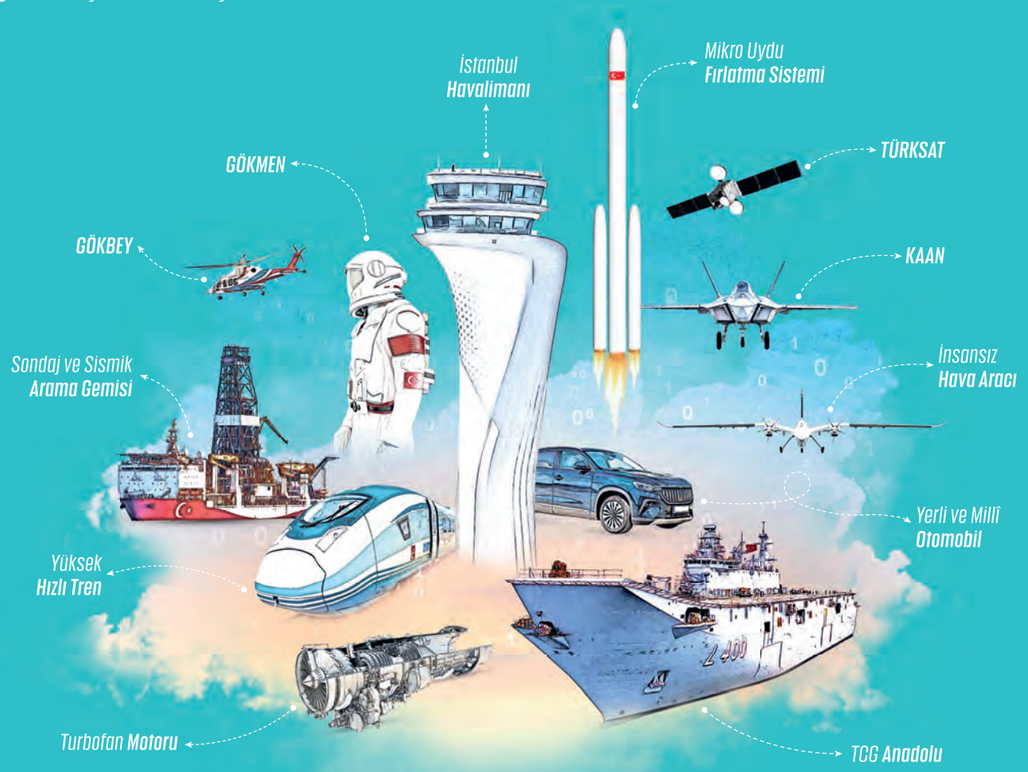
Bu kitaba sığmayan daha neler var!

eba
www.eba.gov.tr

Karekodu okut, bu kitapla ilgili EBA içeriklerine ulaş!

Teknoloji Yolculuğumuz



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

ISBN: 978-975-11-6395-0

Bandrol Uygulamasına İlişkin Usul ve Esaslar Hakkında Yönetmeliğin Beşinci Maddesinin İkinci Fıkrası Çerçevesinde Bandrol Taşıması Zorunlu Değildir.

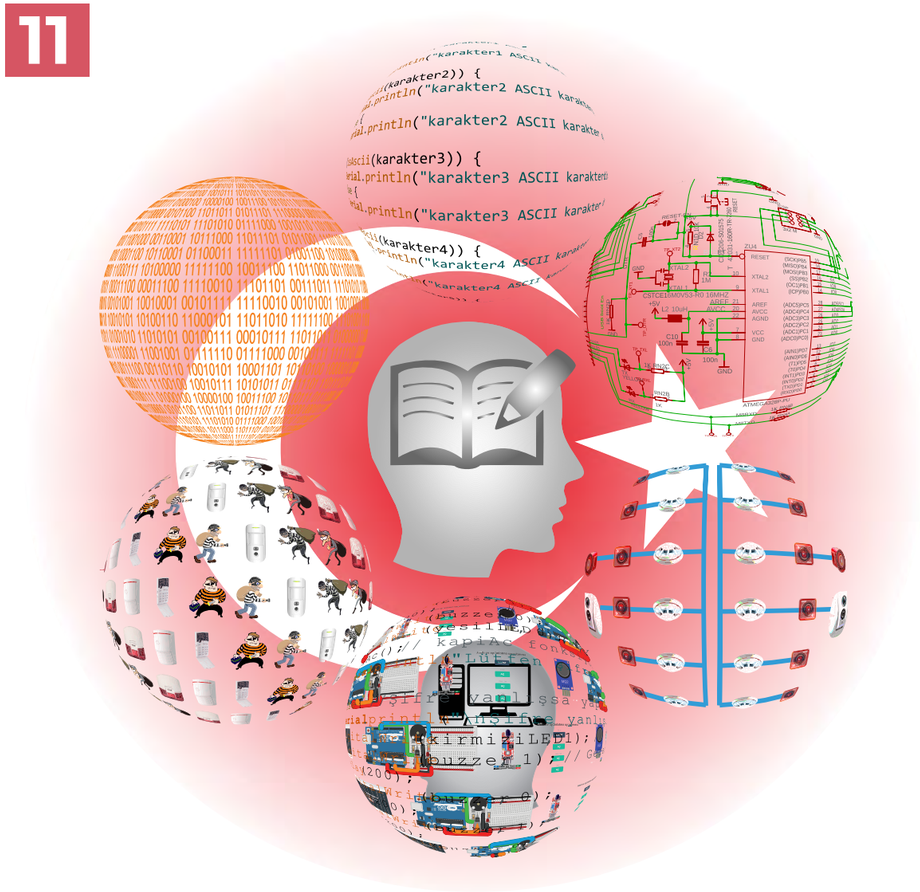
ELEKTRİK-ELEKTRONİK TEKNOLOJİSİ ALANI MİKRODENETLEYİCİLER VE GÜVENLİK ATÖLYESİ 11 DERS MATERYALİ

ELEKTRİK-ELEKTRONİK TEKNOLOJİSİ ALANI

MİKRODENETLEYİCİLER VE GÜVENLİK ATÖLYESİ

DERS MATERYALİ

11



MESLEKİ VE TEKNİK ANADOLU LİSESİ

MESLEKİ VE TEKNİK ANADOLU LİSESİ

ELEKTRİK-ELEKTRONİK TEKNOLOJİSİ ALANI

MİKRODENETLEYİCİLER VE GÜVENLİK ATÖLYESİ

11

DERS MATERYALİ

Yazarlar

Ahmet Zeki AKKAYA
Bekir İŞENGER
Erhan UYTUN
Fatih AKÇAÖZOĞLU
Fatih ÇANKAYA
İsmail SÖNMEZ
Murat ŞAŞAL
Nazım YILDIZ

MİLLÎ EĞİTİM BAKANLIĞI YAYINLARI..... 8361
DERS KİTAPLARI DİZİSİ 2253

Her hakkı saklıdır ve Millî Eğitim Bakanlığına aittir. Ders materyalinin metin, soru şekilleri kısmen de olsa hiçbir surette alınıp yayımlanamaz.

HAZIRLAYANLAR

Dil Uzmanı
Jülide ALTAY

Rehberlik Uzmanı
Feyza SÜNBÜL

Ölçme ve Değerlendirme Uzmanı
Neslihan KOSER

Görsel Tasarım Uzmanı
Firdevs ŞIK

ISBN: 978-975-11-6395-0

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



İSTİKLÂL MARŞI

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

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

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

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

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

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

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

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

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

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

Mehmet Âkif Ersoy

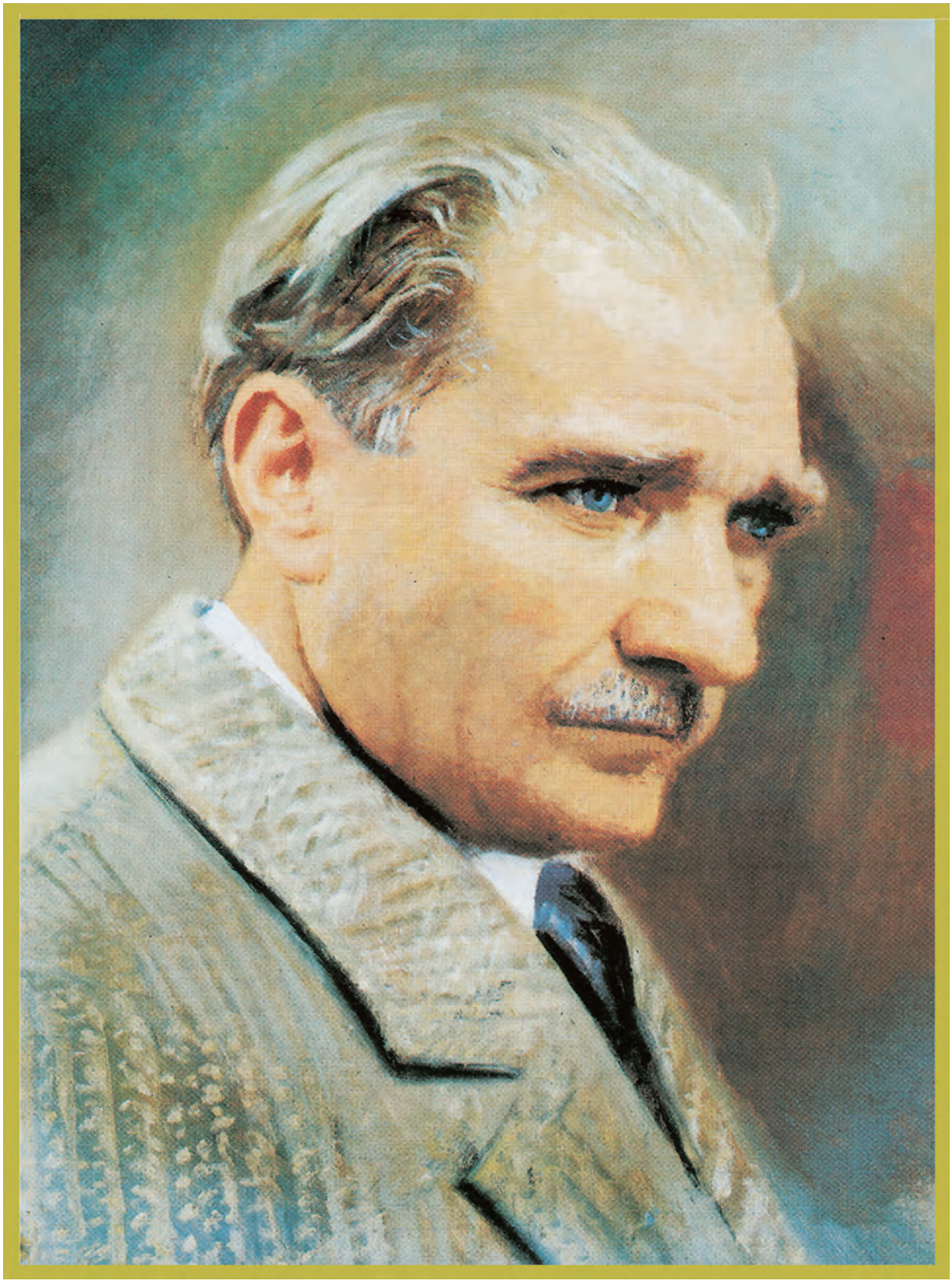
GENÇLİĞE HİTABE

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

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

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

Mustafa Kemal Atatürk



MUSTAFA KEMAL ATATÜRK



İÇİNDEKİLER

DERS MATERYALİNİN TANITIMI **14**

1. ÖĞRENME BİRİMİ **SAYI SİSTEMLERİ VE LOJİK KAPILAR**

1.1. SAYI SİSTEMLERİ **18**
1.1.1. Onluk [Decimal (Desımıl) On Tabanlı] Sayı Sistemi 18
1.1.2. İkilik [Binary (Baynıri) İki Tabanlı] Sayı Sistemi 18
1.1.3. Onaltılık [Hexadecimal (Heksadesımıl) OnaltıTabanlı] Sayı Sistemi 20
1.2. SAYI SİSTEMLERİNİN BİRBİRİNE ÇEVRİLMESİ **21**
1.2.1. Onluk Sayının İkilik Sayı Sistemine Çevrilmesi 21
1.2.2. İkilik Sayının Onluk Sayı Sistemine Çevrilmesi 23
1.2.3. İkilik Sayı Sisteminin Onaltılık Sayı Sistemine Çevrilmesi 24
1.2.4. Onaltılık Sayı Sisteminin İkilik Sayı Sistemine Çevrilmesi 25
1.3. LOJİK ENTEGRELER **25**
1.3.1. TTL [Transistor-Transistor Logic (Transistör-Transistör Mantığı)] Entegreler 26
1.3.2. CMOS [Complementary Metal Oxide Semiconductor (Tamamlayıcı Metal Oksit Yarı İletken)] Entegreler 26
1.3.3. TTL ve CMOS Entegrelerin Karşılaştırılması 27
1.3.4. Entegre Parametreleri 28
1.3.5. Low Aktif, High Aktif Giriş ve Çıkışlar 28
1.4. LOJİK KAPILAR **29**
1.4.1. DEĞİL Kapısı (NOT GATE-INVERTER) 29
DEĞİL Kapısı (NOT GATE) Uygulaması No.: 1 30
1.4.2. VE Kapısı (AND GATE) 31
1.4.3. VEYA Kapısı (OR GATE) 31
VE Kapısı (AND GATE) Uygulaması No.: 2 32
VEYA Kapısı (OR GATE) Uygulaması No.: 3 33
1.4.4. VE DEĞİL Kapısı (NAND GATE) 34
1.4.5. VEYA DEĞİL Kapısı (NOR GATE) 34
VE DEĞİL Kapısı (NAND GATE) Uygulaması No.: 4 35
VEYA DEĞİL Kapısı (NOR GATE) Uygulaması No.: 5 36
1.4.6. ÖZEL VEYA Kapısı (EX-OR GATE) 37
1.4.7. ÖZEL VEYA DEĞİL Kapısı (EX-NOR GATE) 37
Özel Veya Kapısı (Exor Gate) Uygulaması No.: 6 38
Isı (Exnor Gate) Uygulaması No.: 7 39
1.4.8. TAMPON Kapısı (BUFFER GATE) 40
TAMPON Kapısı (BUFFER GATE) Uygulaması No.: 8 41
1.5. LOJİK KAPI UYGULAMALARI **42**
1.5.1. Lojik İfadelerinin Lojik Devrelerle Gösterimi 42
1.5.2. Lojik Devrelerin Lojik İfadelerle Gösterimi 43

2. ÖĞRENME BİRİMİ **MİKRODENETLEYİCİ VE PROGRAMLAMA**

2. 1. MİKROİŞLEMCİLER VE MİKRODENETLEYİCİLER **46**
2.1.2. Mikrodenetleyici 46
2.1.3. Mikroişlemci ve Mikrodenetleyici Arasındaki Farklar 46
2.1.4. Mikrodenetleyici Çeşitleri 46
2.2. MİKRODENETLEYİCİ KARTININ DONANIM YAPISI VE ÖZELLİKLERİ **46**
2.3. MİKRODENETLEYİCİ EDİTÖR PROGRAMI **49**

2.4. MİKRODENETLEYİCİYE PROGRAM YÜKLEME 56
2.5. ALGORİTMA HAZIRLAMA 58
2.6. TEMEL PROGRAMLAMA İŞLEMLERİ 58
2.7. DİJİTAL GİRİŞ ÇIKIŞ İŞLEMLERİ 60
2.7.1. Port Tanımlama pinMode() Fonksiyonu 60
2.7.2. Dijital Çıkış İşlemleri 61
LED'le Dijital Çıkış Uygulaması No.: 1 62
Röleyle Dijital Çıkış Uygulaması No.: 2 67
Trafik Lambası Uygulaması No.: 3 69
Kara Şimşek Uygulaması No.: 4 78
Segment Displayle Dijital Çıkış Uygulaması No.: 5 92
2.7.3. Dijital Giriş İşlemleri 97
Butonla Dijital Giriş Uygulaması No.: 6 98
Reed Switch'le Dijital Giriş Uygulaması No.: 7 98
Optokuplörle Dijital Giriş Uygulaması No.: 8 102
2.7.4. Zaman Gecikmesi Komutları 104
2.7.5. Gelişmiş Giriş Çıkış İşlemleri 105
2.8. SERİ PORT İŞLEMLERİ 114
Seri Port Ekran Uygulaması No.: 9 117
Bilgisayardan Arduino'ya Veri Gönderme No.: 10 121
2.9. ANALOG GİRİŞ ÇIKIŞ İŞLEMLERİ 138
2.9.1. Analog Giriş İşlemleri 138
Potansiyometreyle Analog Giriş Uygulaması No.: 11 139
LDR'yle Analog Giriş Uygulaması No.: 12 141
NTC'yle Analog Giriş Uygulaması No.: 13 143
Alkışla Çalışan LED Uygulaması No.: 14 145
2.9.2. Analog Çıkış İşlemleri 147
Analog Çıkış Uygulaması No.: 15 148
Analog Giriş-Çıkış Uygulaması No.: 16 155
2.10. KESME İŞLEMLERİ 156
Kesme Uygulaması No.: 17 158
2.11. EEPROM İŞLEMLERİ 162
2.11.1. EEPROM Yazma İşlemleri 163
2.11.2. EEPROM Okuma İşlemleri 163

3. ÖĞRENME BİRİMİ MİKRODENETLEYİCİ UYGULAMALARI

3.1. KÜTÜPHANE DOSYASI YÜKLEME 168
3.2. KEYPAD UYGULAMALARI 171
Keypad Uygulaması No.: 1 173
3.3. LCD EKRAN UYGULAMALARI 176
LCD Uygulaması No.: 2 177
3.4. SENSÖR UYGULAMALARI 181
Isı Sensörü Uygulaması No.: 3 182
Işık Sensörü Uygulaması No.: 4 184
Nem Sensörü Uygulaması No.: 5 187
Hareket Sensörü Uygulaması No.: 6 190
Ultrasonik Sensör Uygulaması No.: 7 193
3.5. ELEKTRİK MOTOR UYGULAMALARI 197
DC Motor Uygulaması No.: 8 200
Step Motor Uygulaması No.: 9 205

Servo Motor Uygulaması No.: 10 209
3.6. HABERLEŞME UYGULAMALARI 211
Bluetooth Uygulaması No.: 11 217
IR ile Uzaktan Kumanda Uygulaması No.: 12 220
RFID Uygulaması No.: 13 224
Wi-Fi Uygulaması No.: 14 242
Engelden Kaçan Robot Uygulaması No.: 15 244
Bluetooth Kontrollü Araç Uygulaması No.: 16 249
Uzaktan Kontrollü Araba Uygulaması No.: 17 254
Elle Uzaktan Kontrollü Araba Uygulaması No.: 18 264
Çizgi İzleyen Robot Uygulaması No.: 19 272
Wi-Fi Kontrollü Araba Uygulaması No.: 20 280

4. ÖĞRENME BİRİMİ YANGIN ALARM SİSTEMLERİ

4. YANGIN ALARM SİSTEMLERİ 290
4.1. YANGIN ALARM SANTRALİ 291
4.1.1. Konvansiyonel Sistem Yangın Alarm Santrali 293
4.1.2. Adreslemeli Sistem Yangın Alarm Santrali 296
4.2. YANGIN ALARM SİSTEMİ DONANIMLARI 308
4.2.1. Duman Dedektörü 304
4.2.2. Isı (Sıcaklık) Dedektörü 310
4.2.3. Multi (Kombine) Dedektör 311
4.2.4. Alev Dedektörü 312
4.2.5. Dedektör Taban Ayağı ve Bağlantıları 312
4.2.6. Yangın İhbar Butonu 314
4.2.7. Siren 314
4.2.8. Paralel İhbar Lambası 315
4.2.9. Doğal Gaz–LPG (Likit Petrol Gazı) Dedektörü 316
4.2.10. Acil Çıkış Yönlendirme Levhaları 318
4.2.11. GSM Modülü 318
4.2.12. Karbonmonoksit (CO) Alarm Santrali ve Gaz Dedektörü 320
4.2.13. Otomatik Yangın Söndürme Sistemleri 322
4.2.14. Yangın Alarm Sistemlerinde Kullanılan Yardımcı Ürünler 324
4.2.14.1. Adresli Siren Kontrol Modülü 324
4.2.14.2. Adresli Röle Modülü 324
4.2.14.3. Adresli Kontak İzleme Modülü 324
4.2.14.4. Adresli Zone Modülü 325
4.2.14.5. İzolatör Modülü 325
4.2.14.6. Termal Printer 325
4.2.14.7. Network Haberleşme Kartı 326
Konvansiyonel Yangın Alarm Santralinin Testi No.: 1 327
Konvansiyonel Yangın Alarm Santralinde Ölçüm No.: 2 329
Konvansiyonel Yangın Alarm Santraline Dedektör Bağlantısı No.: 3 332
Konvansiyonel Yangın Alarm Santralinde Kullanılan Dedektörlerin Akım ve Gerilimlerinin Ölçülmesi No.: 4 .. 334
Konvansiyonel Yangın Alarm Santraline Yangın İhbar Butonu Bağlanması No.: 5 336
Konvansiyonel Yangın Alarm Santralinde Dedektörlere Paralel İhbar Lambasının Bağlantısı No.: 6 338
Konvansiyonel Yangın Alarm Santraline Siren Bağlanması No.: 7 340
Konvansiyonel Yangın Alarm Santralinde 2 Zone Kullanılarak Dedektör, Yangın İhbar Butonu, Paralel İhbar Lambası ve Siren Bağlanması No.: 8 342
Konvansiyonel Yangın Alarm Santraline Doğal Gaz-LPG Dedektörü Bağlanması No.: 9 343

Konvansiyonel Yangın Alarm Santraline Bütün Ürünlerin Bağlantısının Yapılması No.: 10 345
Konvansiyonel Yangın Alarm Santralinde GSM Modülü Kullanımı No.: 11 346
Konvansiyonel Yangın Alarm Santraline DC 24 V Fan ve AC 220 V Lamba Bağlantısı Yapılması No.: 12 .. 347
Konvansiyonel Yangın Alarm Santraline Selenoid Valf ve Akıllı Kilit Bağlantısı Yapılması No.: 13. 348
Yangın İhbar Butonu Yapımı No.: 14 349

5. ÖĞRENME BİRİMİ HIRSIZ ALARM SİSTEMLERİ

5. HIRSIZ ALARM SİSTEMLERİ 352
5.1. HIRSIZ ALARM SANTRALİ 353
5.1.1. Kablolu Sistem Hırsız Alarm Santrali 355
5.1.2. Kablosuz Sistem Hırsız Alarm Santrali 361
5.2. HIRSIZ ALARM SİSTEMİ DONANIMLARI 363
5.2.1. Keypad 363
5.2.2. PIR (Pasif İnfrared) Dedektörü 365
5.2.3. Cam Kırılma Dedektörü 368
5.2.4. Darbe Dedektörleri 369
5.2.5. Su Basma Dedektörleri 369
5.2.6. Dedektör Bağlantıları 370
5.2.7. Beam (Işın) Dedektörleri 373
5.2.8. Hırsız Alarm Sistemlerinde Yangın Dedektörlerinin Kullanımı 375
5.2.9. Manyetik Kontak 375
5.2.10. Siren 376
5.2.11. Hırsız Alarm Sistemlerinde Kullanılan Buton Çeşitleri 377
5.2.12. Kablosuz Hırsız Alarm Sistemlerinde Kullanılan Ürünler 379
5.2.13. Alarm İzleme ve Kontrol 380
5.2.14. Hırsız Alarm Sistemlerinde Kullanılan Yardımcı Ürünler 382
5.2.14.1. Uzaktan Kumanda Modülü 382
5.2.14.2. Kablolu-Kablosuz Zone Genişleme Modülleri 382
5.2.14.3. LAN (Network) Modülü – İnternet İletişim Modülü 382
5.2.14.4. GSM/GPRS Haberleşme Modülleri 383
5.3. HIRSIZ ALARM SİSTEMİNİN PROGRAMLANMASI 383
5.3.1. Keypadin Tanıtılması 384
5.3.2. Fabrikasyon Şifreler 384
5.3.3. Zone Programlanması 385
5.3.3.1. Zone Program Menüsüne Girilmesi 385
5.3.3.2. Zone Numarasının Girilmesi 385
5.3.3.3. Zone Tipinin Belirlenmesi 386
5.3.3.4. Zone Kısımlandırılması 387
5.3.3.5. Zone Özellikleri 387
5.3.3.6. Zone Onayı ve Zone Programından Çıkış 387
5.3.3.7. Zone Programlama Örnekleri 388
5.3.4. Tarih ve Saat Programlama 388
5.3.5. Sürelerin Programlanması 388
5.3.6. PGM Programlama 389
5.3.7. Zamanlı Otomatik Kurulumun Programlanması 390
5.3.8. Kablosuz Ürünlerin Programlanması 390
5.3.9. Uzaktan Kumandanın Programlanması 391
5.3.10. Kullanıcı Şifrelerinin Programlanması 391
5.3.11. Haber Alma Merkezinin Programlaması 392
5.3.12. İOS-Android Yazılımların Programlanması 392

Hırsız Alarm Sistemlerinde Kullanılan Zone Bağlantı Modellemeleri No.: 1 394
Hırsız Alarm Sistemlerinde Kullanılan Dedektörler No.: 2 396
Hırsız Alarm Sistemlerinde Kullanılan Dedektörlerle Lamba Uygulaması No.: 3 399
Kablolu Hırsız Alarm Santralinin Resetlenmesi ve Tarih ile Saat Ayarı No.: 4 401
Kablolu Hırsız Alarm Santralinde Zone Programlama ve Zone Ölçümleri No.: 5 403
Kablolu Hırsız Alarm Santralinde Siren Bağlantısı ve Ölçümleri No.: 6 406
Üç Dedektör ve Bir Siren Bağlantılı Kablolu Hırsız Alarm Sistemi Kurulumu No.: 7 409
Bir Zonea Birden Fazla Dedektörü Seri Bağlama 411
Double Zone Bağlantısı No.: 9 413
Yangın Alarm Dedektörlerinin ve PGM Çıkışlarının Kullanımı No.: 10 415
Kablosuz Hırsız Alarm Santralinin Tanıtılması ve Programlanması No.: 11....417
Kablosuz Hırsız Alarm Santralinin Programlanması No.: 12 420

6. ÖĞRENME BİRİMİ ELEKTRONİK GÜVENLİK DEVRELERİ

Sıcaklık Dedektörü Devresi No.: 1 422
Alev Dedektörü Devresi No.: 2 424
Gaz Dedektörü Devresi No.: 3 426
Hareket Dedektörü Devresi No.: 4 428
Yaklaşım Dedektörü Devresi No.: 5 430
Şifreli Kilit Uygulaması No.: 6 432
Kartlı Geçiş Uygulaması No.: 7 436
Parmak İzi Okuma Uygulaması No.: 8 440
Akıllı Ev Aydınlatma Uygulaması No.: 9 447
Akıllı Ev Isıtma Uygulaması No.: 10 458
KAYNAKÇA 460

Öğrenme biriminin numarasını gösterir.

Öğrenme biriminin kapağını gösterir.

Öğrenme biriminin görselini gösterir.

Öğrenme biriminin ismini gösterir.

Öğrenme biriminindeki konu başlıklarını gösterir.

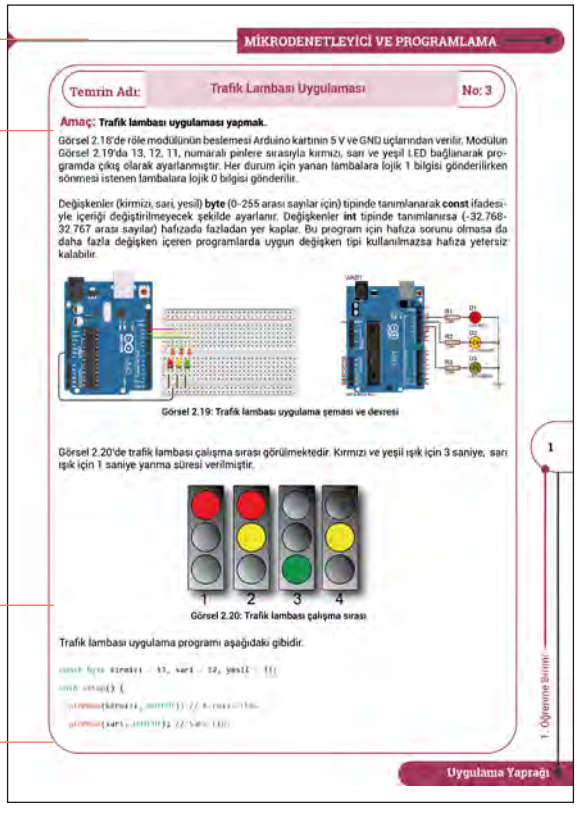


Öğrenme biriminin ismini gösterir.

Uygulamanın amacını gösterir.

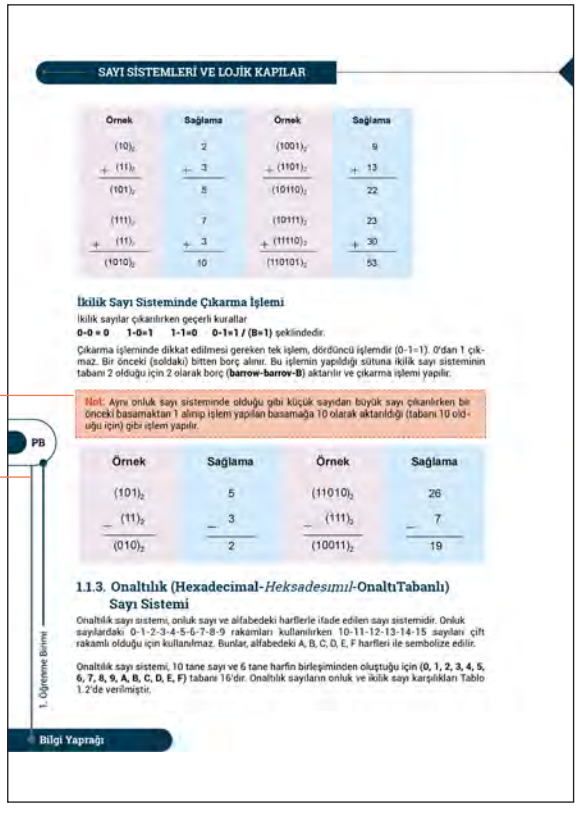
Görsel numarasını gösterir.

Uygulama yaprağında olduğunuzu gösterir.



Öğrenme birimindeki önemli not kısımlarını gösterir.

Öğrenme biriminin sayfa numarasını gösterir.



SAYI SİSTEMLERİ VE LOJİK KAPILAR

Örnek	Sağlama	Örnek	Sağlama
$(10)_2$ + $(11)_2$ = $(101)_2$	2 + 3 = 5	$(1001)_2$ + $(1101)_2$ = $(10110)_2$	9 + 13 = 22
$(111)_2$ + $(11)_2$ = $(1010)_2$	7 + 3 = 10	$(10111)_2$ + $(11110)_2$ = $(110101)_2$	23 + 30 = 53

İkilik Sayı Sisteminde Çıkarma İşlemi

İkilik sayılar çıkarılırken geçerli kurallar

0-0 = 0 1-0=1 1-1=0 0-1=1 / (B=1) şeklindedir.

Çıkarma işleminde dikkat edilmesi gereken tek işlem, dördüncü işlemdir (0-1=1). 0'dan 1 çıkmaz. Bir önceki (soldaki) bitten borç alınır. Bu işlemin yapıldığı sütuna ikilik sayı sisteminin tabanı 2 olduğu için 2 olarak borç (**barrow-barrov-B**) aktarılır ve çıkarma işlemi yapılır.

Not: Aynı onluk sayı sisteminde olduğu gibi küçük sayıdan büyük sayı çıkarılırken bir önceki basamaktan 1 alınıp işlem yapılan basamağa 10 olarak aktarıldığı (tabanı 10 olduğu için) gibi işlem yapılır.

PB

Örnek	Sağlama	Örnek	Sağlama
$(101)_2$ − $(11)_2$ = $(010)_2$	5 − 3 = 2	$(11010)_2$ − $(111)_2$ = $(10011)_2$	26 − 7 = 19

1.1.3. Onaltılık (Hexadecimal-*Heksadesımıl*-OnaltıTabanlı) Sayı Sistemi

Onaltılık sayı sistemi, onluk sayı ve alfabedeki harflerle ifade edilen sayı sistemidir. Onluk sayılardaki 0-1-2-3-4-5-6-7-8-9 rakamları kullanılırken 10-11-12-13-14-15 sayıları çift rakamlı olduğu için kullanılmaz. Bunlar, alfabedeki A, B, C, D, E, F harfleri ile sembolize edilir.

Onaltılık sayı sistemi, 10 tane sayı ve 6 tane harfin birleşiminden oluştuğu için **(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)** tabanı 16'dır. Onaltılık sayıların onluk ve ikilik sayı karşılıkları Tablo 1.2'de verilmiştir.

1. Öğrenme Birimi

Bilgi Yaprağı

Öğrenme biriminin konu başlıklarını gösterir.

Öğrenme birimindeki görselleri gösterir.



MİKRODENETLEYİCİ VE PROGRAMLAMA

2.5. ALGORİTMA HAZIRLAMA

Algoritma, bir problemi çözmek için takip edilecek sonlu sayıda adımdan oluşan bir çözüm yoludur. Diğer bir ifadeyle algoritma, bir problemin mantıksal çözümünün adım adım nasıl gerçekleştirileceğinin sözlü ifadesidir.

Algoritmayla oluşturulan çözümler sözel olarak ifade edildiğinden algoritmayı herkesin aynı sonucu çıkarabileceği hâle getirmek için akış diyagramları kullanılır. Akış diyagramları sembollerden oluşmaktadır. Her sembolün belli bir işlevi vardır.

Algoritması oluşturulmuş bir problemin bilgisayar ortamına aktarılmış hâline **program** denir. Program, problemin çözümünde yapılması gereken işlemler bütününün kod karşılığıdır. Algoritmaların program hâline getirilmesi için programlama dilleri kullanılır. Programlama dilleri kullanılarak yazılımlar geliştirilir. Arduino programlama dili C/C++ programlama dilinden üretilmiş bir dildir.

Görsel 2.15'te butona basıldığında LED'in yandığı bir olayın algoritması, akış diyagramı ve programı görülmektedir.

Görsel 2.15: Karar yapısı algoritması

2.6. TEMEL PROGRAMLAMA İŞLEMLERİ

Arduino programlama dili, C programlama diline benzemektedir. Her programlama dilinde yapılan temel işlemler Arduino programlama dilinde de yapılmaktadır.

Arduino IDE'de yeni bir taslak oluşturulduğunda taslak (sketch) Görsel 2.16'da görüldüğü gibi iki hazır fonksiyonla gelmektedir. Bunlar setup() (ayar) ve loop() (döngü) fonksiyonlarıdır. Bu fonksiyonlar taslakta bulunmazsa derleyici hata verir.

PB

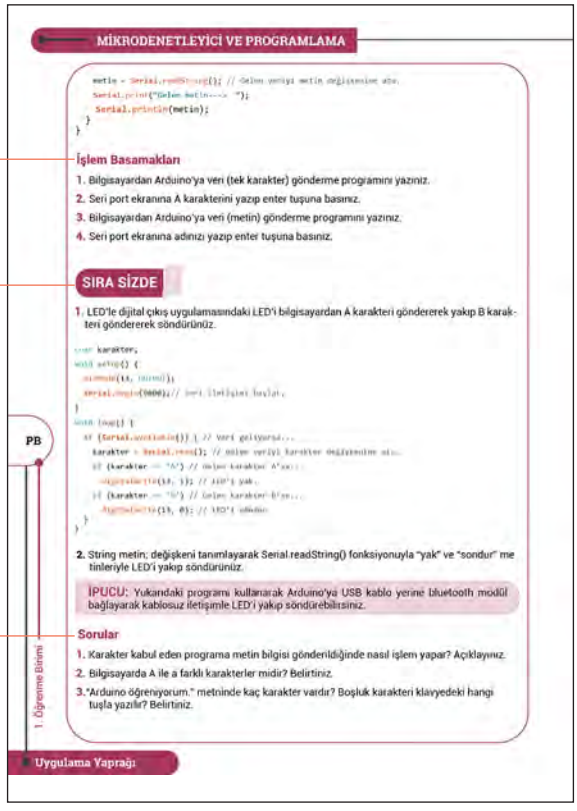
1. Öğrenme Birimi

Bilgi Yaprağı

Öğrenme biriminin sonundaki işlem basamaklarını gösterir.

Uygulama yaprağındaki sıra sizde bölümünü gösterir.

Uygulama yaprağında ipucu ve sorularda olduğunuzu gösterir.

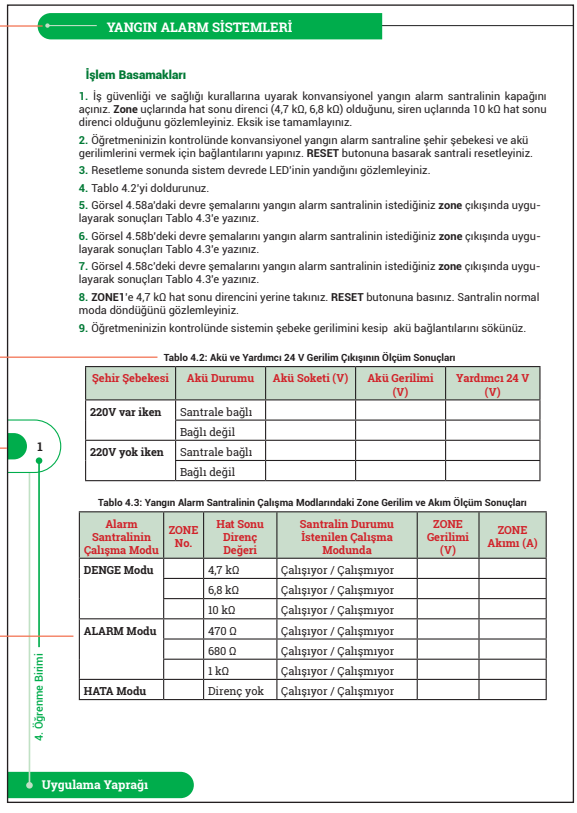


Öğrenme biriminin ismini gösterir.

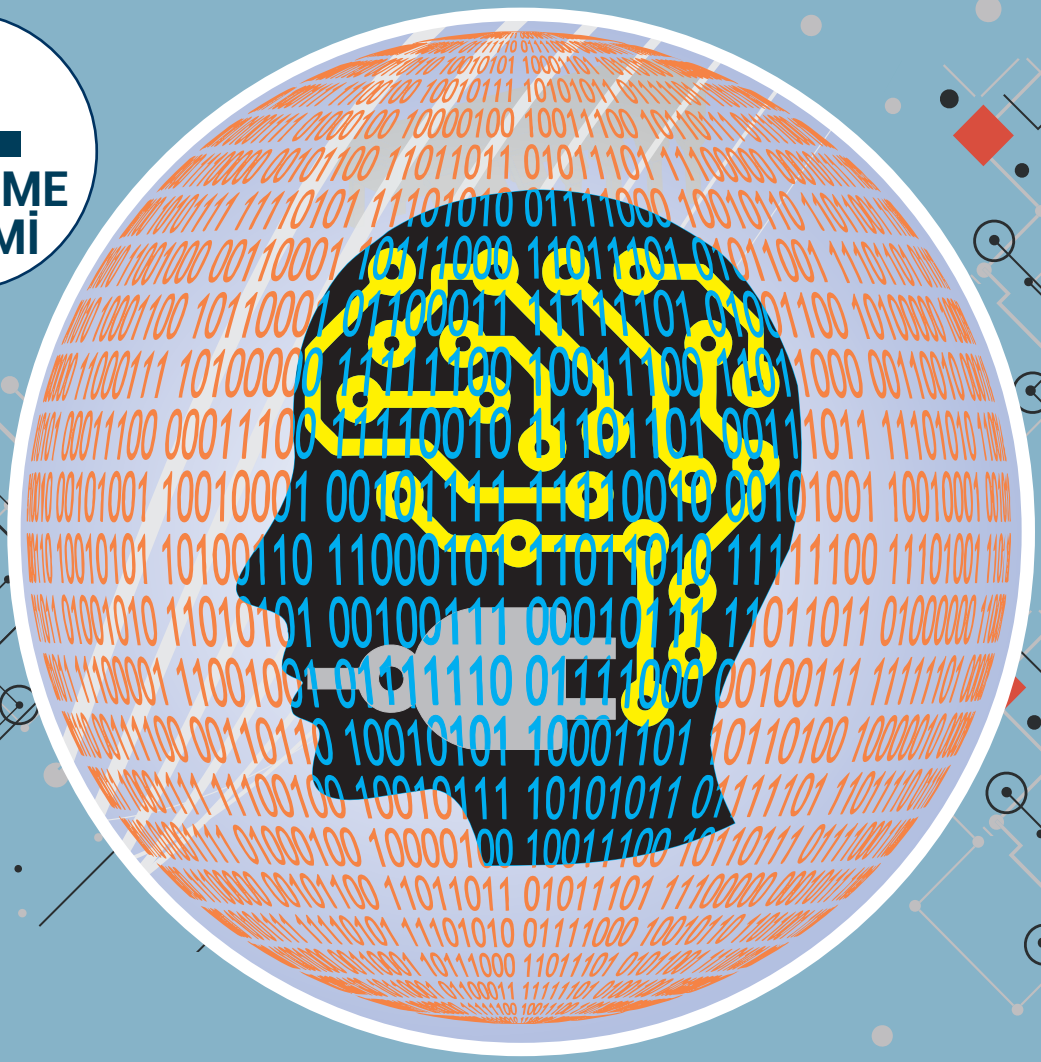
Tablonun başlık numarasını gösterir.

Sayfa numarasını gösterir.

Öğrenme birimi içindeki tabloları gösterir.



1. ÖĞRENME BİRİMİ



SAYI SİSTEMLERİ VE LOJİK KAPILAR

NELER ÖĞRENECEKSİNİZ ?

- Sayı Sistemleri
- Sayı Sistemlerinin Birbirine Çevrilmesi
- Lojik Entegreler
- Lojik Kapılar
- Lojik Kapı Uygulamaları

TEMEL KAVRAMLAR

double word byte

bit

nibble

1.1. SAYI SİSTEMLERİ

Dijital elektronik devreleri ile tasarım ve üretim yapılması veya yapılanın anlaşılması için temel matematik kuralları ve sayı sistemlerinin bilinmesi gerekir. Yaşamda onluk sayı sistemi, dijital elektronikte ise ikilik ve onaltılık sayı sistemleri kullanılır.

1.1.1. Onluk [Decimal (Desımıl) On Tabanlı] Sayı Sistemi

Günlük hayatta kullanılan sayı sistemine **onluk sayı sistemi** denir. Onluk sayı sisteminde **(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)** olmak üzere toplam on tane rakam kullanılır. Kullanılan rakamlardan en küçüğü **0**'dır. En büyüğü ise taban sayının 1 eksiği olan **9** rakamıdır.
On tabanlı sayıların basamak değerleri aşağıdaki gibi gösterilir:

ÖRNEK
$(254)_{10}$ sayısını onluk tabanda göstererek bulunuz.

ÇÖZÜM
$(254)_{10} = 2.10^2 + 5.10^1 + 4.10^0$
$= 2.(100) + 5.(10) + 4.(1)$
$= 200 + 50 + 4$

ÖRNEK
$(2103)_{10}$ sayısını onluk tabanda göstererek bulunuz.

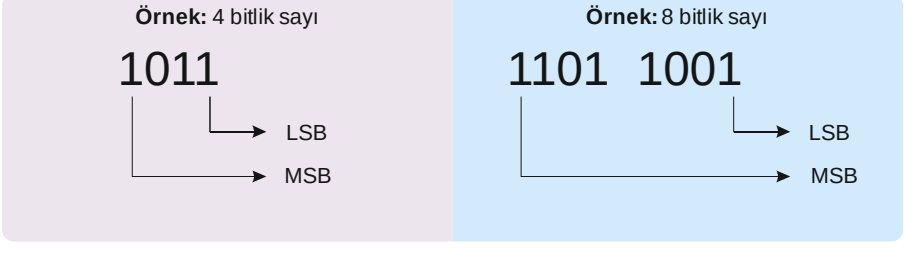
ÇÖZÜM
$(2103)_{10} = 2.(10^3) + 1.(10^2) + 0.(10^1) + 3.(10^0)$
$= 2.(1000) + 1.(100) + 0.(10) + 3.(1)$
$= 2000 + 100 + 0 + 3$

Onluk sayı sisteminde dört işlem (toplama, çıkarma, çarpma ve bölme) günlük yaşamda kullanıldığı gibi yapılır.

1.1.2. İkilik [Binary (Baynıri) İki Tabanlı] Sayı Sistemi

İkilik sayı sistemi dijital elektronikte en çok kullanılan sayı sistemidir ve dijital elektroniğin temelidir. İkilik sayı sistemi 0 ve 1 rakamlarının kullanıldığı sistemdir. Bu sayı sisteminde iki rakam olduğu için taban 2'dir.

İkilik sayılar yazılırken en sağdaki basamağa en düşük değerlikli bit [Least Significant Bit (List Sıgnifikınt Bit) **LSB**], en soldaki basamağa en yüksek değerlikli bit [Most Significant Bit (Mos Sıgnifikınt Bit) **MSB**] adı verilir.

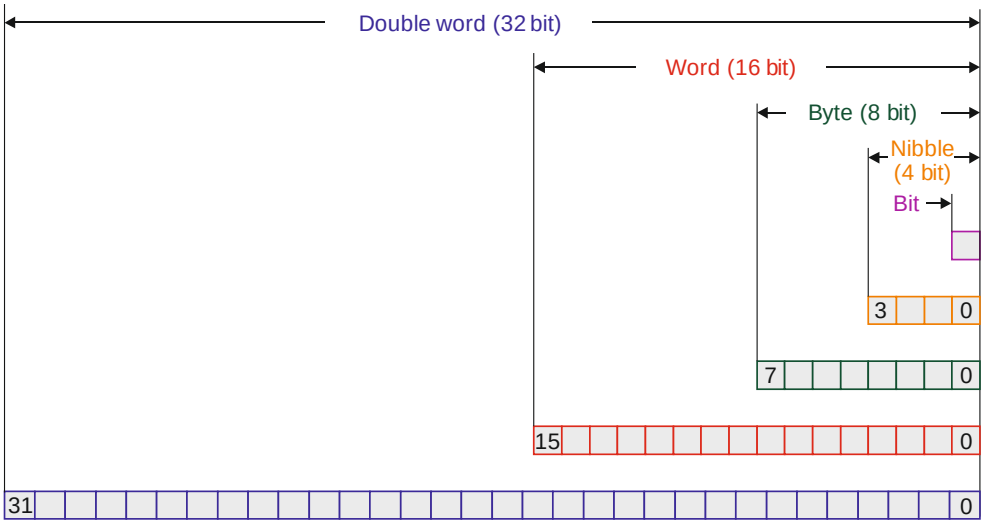


Dört **bit**lik ikilik sayıda bit ağırlıkları Tablo 1.1'de gösterilmiştir.

Tablo 1.1: Dört Bitlik İkili Sayının Bit Ağırlıkları

	4. Basamak	3. Basamak	2. Basamak	1. Basamak
Üst değeri	2^3	2^2	2^1	2^0
Ağırlık	8	4	2	1

İkilik sayı sistemindeki her bir rakama (0 veya 1) **bit** denir. Bit, veri saklamanın en küçük birimidir. 8 bitin yan yana gelmesiyle oluşan veri tipine **byte** (bayt) denir. 4 bitin yan yana gelmesiyle oluşan veri tipine veya 1/2 **byte**a **nibble** (nibıl) denir. 16 bitin yan yana gelmesiyle oluşan veri tipine veya 2 byte bilgiye **word** (vörd) denir. 32 bitin yan yana gelmesiyle oluşan veri tipine veya 2 word bilgiye veya 4 byte bilgiye **double word** (dabıl vörd) denir. Görsel 1.1'de dijital elektronikte kullanılan veri tipleri görülmektedir.



Görsel 1.1: Veri tipleri

İkilik Sayı Sisteminde Toplama İşlemi

İkilik sayılar toplanırken geçerli kurallar
0+0=0 0+1=1 1+0=1 1+1=10 / (**C=1**) şeklindedir.

Burada dikkat edilmesi gereken tek işlem, dördüncü işlemdir (1+1=10). İkilik sayı sisteminde 0 ve 1 dışında rakam kullanılmadığı için (1+1=2) yazılamaz. Toplamda çıkan 2 yerine soldaki basamağa 1 elde eklenir ve toplam olarak da 0 yazılır. Elde sözcüğü İngilizcede **carry** (keri) ile ifade edildiğinden lojik devre işlemlerinde ilk harfiyle (**C**) gösterilir.

Örnek	Sağlama	Örnek	Sağlama
$(10)_2 + (11)_2 = (101)_2$	$2 + 3 = 5$	$(1001)_2 + (1101)_2 = (10110)_2$	$9 + 13 = 22$
$(111)_2 + (11)_2 = (1010)_2$	$7 + 3 = 10$	$(10111)_2 + (11110)_2 = (110101)_2$	$23 + 30 = 53$

İkilik Sayı Sisteminde Çıkarma İşlemi

İkilik sayılar çıkarılırken geçerli kurallar

0-0 = 0 **1-0=1** **1-1=0** **0-1=1 / (B=1)** şeklindedir.

Çıkarma işleminde dikkat edilmesi gereken tek işlem, dördüncü işlemdir (0-1=1). 0'dan 1 çıkmaz. Bir önceki (soldaki) bitten borç alınır. Bu işlemin yapıldığı sütuna ikilik sayı sisteminin tabanı 2 olduğu için 2 olarak borç [**barrow (**barrov**) B**] aktarılır ve çıkarma işlemi yapılır.

NOT

Onluk sayı sisteminde olduğu gibi küçük sayıdan büyük sayı çıkarılırken bir önceki basamaktan 1 alınıp işlem yapılan basamağa 10 olarak aktarıldığı (tabanı 10 olduğu için) gibi işlem yapılır.

Örnek	Sağlama	Örnek	Sağlama
$(101)_2 - (11)_2 = (010)_2$	$5 - 3 = 2$	$(11010)_2 - (111)_2 = (10011)_2$	$26 - 7 = 19$

1.1.3. Onaltılık [Hexadecimal (Heksadesımıl) Onaltı Tabanlı] Sayı Sistemi

Onaltılık sayı sistemi, onluk sayı ve alfabedeki harflerle ifade edilen sayı sistemidir. Onluk sayılardaki 0-1-2-3-4-5-6-7-8-9 rakamları kullanılırken 10-11-12-13-14-15 sayıları çift rakamlı olduğu için kullanılmaz. Bunlar, alfabedeki A, B, C, D, E, F harfleri ile sembolize edilir.

Onaltılık sayı sistemi, 10 tane sayı ve 6 tane harfin birleşiminden oluştuğu için (**0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)** tabanı 16'dır. Onaltılık sayıların onluk ve ikilik sayı karşılıkları Tablo 1.2'de verilmiştir.

Tablo 1.2: Onaltılık Sayıların Onluk ve İkilik Sayı Karşılıkları

Onaltılık sayılar	İkilik sayılar	Onluk sayılar	Onaltılık sayılar	İkilik sayılar	Onluk sayılar
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	A	1010	10
3	0011	3	B	1011	11
4	0100	4	C	1100	12
5	0101	5	D	1101	13
6	0110	6	E	1110	14
7	0111	7	F	1111	15

1.2. SAYI SİSTEMLERİNİN BİRBİRİNE ÇEVRİLMESİ

Günlük hayatta onluk, sayısal elektronikte ise ikilik ve onaltılık sayı sistemi kullanılmaktadır. Sayı sistemlerinin birbirine çevrilmesine sıkça rastlanır.

1.2.1. Onluk Sayının İkilik Sayı Sistemine Çevrilmesi

10 tabanlı bir sayıyı ikilik sayı tabanına dönüştürmek için birkaç yöntem vardır. Burada sadece iki yönteme yer verilmiştir.

1.Yöntem: Onluk sayılar, ikilik sayıya çevrilirken 2'ye bölme yöntemi kullanılır. Çevrilmek istenen sayı, bölüm değeri 2'den küçük olana kadar 2'ye bölünür. 2'den küçük olan bölüm ile başlanarak sırayla **sondan başa doğru** kalanlar yazılır. Elde edilen 1 ve 0'larla oluşan sayı, onluk sayının ikilik karşılığıdır.

Örnek: $(6)_{10}$ sayısını ikilik sayıya çeviriniz.

Çözüm

Bölünen	Bölüm	Kalan	Yazılış yönü	Bit ağırlığı
6:2	3	0	↑	LSB
3:2	1	1		
1:2	1	1		MSB

$(6)_{10}=(110)_2$

Örnek: $(17)_{10}$ sayısını ikilik sayıya çeviriniz.

Çözüm

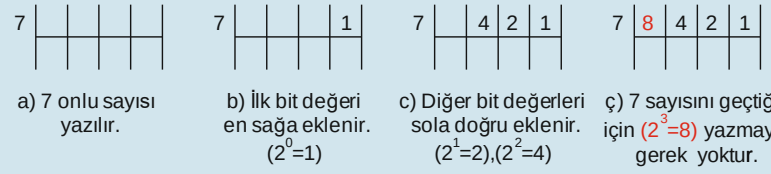
Bölünen	Bölüm	Kalan	Yazılış yönü	Bit ağırlığı
17:2	8	1	↑	LSB
8:2	4	0		
4:2	2	0		
2:2	1	0		
1:2	0	1		MSB

$(17)_{10}=(10001)_2$

2.Yöntem: Onluk sayı ikilik sayıya çevrilirken onluk sayı sola yazılır. Biraz boşluk bırakılarak sağdan sola doğru onluk sayıya en yakın ikilik sayıya ulaşılana kadar 2'nin kuvveti olan sayılar yazılır. Daha sonra onluk sayı, en yüksek basamak değerinden başlamak şartı ile 2'nin kuvvetlerine, kalan sıfır olana kadar bölünür. Tabloda elde edilen sayılar onluk sayının ikilik sayı sistemindeki karşılığıdır.

ÖRNEK: $(7)_{10}$ sayısını ikilik sayı sistemine çeviriniz.

ÇÖZÜM: 7 sayısı ikilik sayı sistemine çevrilirken 2. yöntem kullanılacaktır. İşlem basamakları

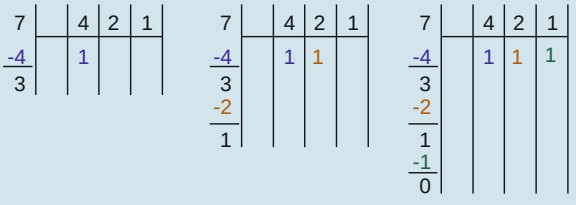


a) 7 onlu sayısı yazılır.

b) İlk bit değeri en sağa eklenir. (2^0=1)

c) Diğer bit değerleri sola doğru eklenir. (2^1=2),(2^2=4)

ç) 7 sayısını geçtiği için (2^3=8) yazmaya gerek yoktur.



İşlemin sonucu $(7)_{10}=(111)_2$

d) 7 sayısı 0 veya 1 olana kadar bölme işlemi yapılır.

ÖRNEK: $(9)_{10}$ sayısını ikilik sayıya çeviriniz.

ÇÖZÜM:

9	8	4	2	1
-8	1	0	0	1
1				
-1				
0				

Sonuç:$(1001)_2$

ÖRNEK: $(18)_{10}$ sayısını ikilik sayıya çeviriniz.

ÇÖZÜM:

18	16	8	4	2	1
-16	1	0	0	1	0
2					
-2					
0					

Sonuç:$(10010)_2$

ÖRNEK: $(25)_{10}$ sayısını ikilik sayıya çeviriniz.

ÇÖZÜM:

25	16	8	4	2	1
-16	1	1	0	0	1
9					
-8					
1					

Sonuç:$(11001)_2$

ÖRNEK: $(47)_{10}$ sayısını ikilik sayıya çeviriniz.

ÇÖZÜM:

47	32	16	8	4	2	1
-32	1	0	1	1	1	1
15						
-8						
7						
-4						
3						
-2						
1						
-1						
0						

Sonuç:$(101111)_2$

1.2.2. İkilik Sayının Onluk Sayı Sistemine Çevrilmesi

1.Yöntem: İkilik sayıların onluk sayı sistemindeki değerini bulabilmek için basamaklar sağdan sola doğru 2'nin kuvveti ile (2^0, 2^1, 2^2,2^3, 2^4….) çarpılır. Çarpım sonucu bulunan değerler toplanarak ikili sayının onluk karşılığı elde edilir.

ÖRNEK: $(100)_2$ sayısını onluk sayıya çeviriniz.

ÇÖZÜM:

$(100)_2$

2^0
2^1
2^2

$=1x2^2+0x2^1+0x2^0$
$=1x4+0x2+0x1$
$=4+0+0$
$=4$

ÖRNEK: $(1011)_2$ sayısını onluk sayıya çeviriniz.

ÇÖZÜM:

$(1011)_2$

2^0
2^1
2^2
2^3

$=1x2^3+0x2^2+1x2^1+1x2^0$
$=1x8+0x4+1x2+1x1$
$=8+0+2+1$
$=11$

2.Yöntem: İkilik sayı sistemindeki sayı, onluk sayı sistemine çevrilirken öncelikle verilen ikilik sayı aralıklı olarak yazılır. Aşağıda verilen işlem basamakları takip edilir. Tablonun üstüne basamak ağırlıkları yazılır. Her bir sütun kendi içinde çarpılır. Çarpım sonuçları toplanarak onluk sayı elde edilir.

ÖRNEK: $(1001)_2$ sayısını onluk sayıya çeviriniz.

ÇÖZÜM:

1	0	0	1

İkilik sayı tabloda alta yazılır.

8	4	2	1
1	0	0	1

Basamak ağırlık değeri üste yazılır.

8	4	2	1
1	0	0	1

×

8+0+0+1

Her sütun kendi içinde çarpılır.

8	4	2	1
1	0	0	1

Sonuç= $(9)_{10}$

Çarpımların sonucu toplanır.

ÖRNEK: $(1011)_2$ sayısını onluk sayıya çeviriniz.

ÇÖZÜM:

8	4	2	1
1	0	1	1

×

8+0+2+1

$(11)_{10}$

ÖRNEK: $(10101)_2$ sayısını onluk sayıya çeviriniz.

ÇÖZÜM:

16	8	4	2	1
1	0	1	0	1

×

16+0+4+0+1

$(21)_{10}$

ÖRNEK: $(11010)_2$ sayısını onluk sayıya çeviriniz.

ÇÖZÜM:

16	8	4	2	1
1	1	0	1	0

×

16+8+0+2+0

$(26)_{10}$

ÖRNEK: $(101101)_2$ sayısını onluk sayıya çeviriniz.

ÇÖZÜM:

32	16	8	4	2	1
1	0	1	1	0	1

×

32+0+8+4+0+1

$(45)_{10}$

1.2.3. İkilik Sayı Sisteminin Onaltılık Sayı Sistemine Çevrilmesi

16 sayısı 2'nin 4'üncü kuvveti olduğu için ikilik sayı dörder basamaktan oluşan gruplara ayrılır. Her dörtlü gruptaki ikilik sayı onluk sayı sistemine çevriliyormuş gibi işlem yapılır.

Gruplandırma işlemi, en sağdan başlanarak sola doğru yapılır. En sondaki grupta dörtlü oluşmuyorsa sola doğru sıfır eklenerek grup sayısı 4'e tamamlanır. Çünkü soldaki sıfırların sayı değerine etkisi yoktur. Gruptaki sayıların karşılıkları olan sayılar soldan sağa doğru yazılınca onaltılık sayı bulunmuş olur.

ÖRNEK: $(100111111010111)_2$ sayısını onaltılık sayı sistemine çeviriniz.

ÇÖZÜM: İkilik sayı sağdan sola doğru dörder bitlik gruplara ayrılır. En sol grup dört bite tamamlanmamışsa soluna 0 eklenerek dört bite tamamlanır.

0 1 0 0	1 1 1 1	1 1 0 1	0 1 1 1
4	15	13	7
4	F	D	7

Sonuç= $(4FD7)_{16}$

ÖRNEK : $(1001101101011101)_2$ sayısını onaltılık sayı sistemine çeviriniz.

ÇÖZÜM:

1 0 0 1	1 0 1 1	0 1 0 1	1 1 0 1
9	11	5	13
9	B	5	D

Sonuç= $(9B5D)_{16}$

1.2.4. Onaltılık Sayı Sisteminin İkilik Sayı Sistemine Çevrilmesi

İkilik sayının onaltılık sayı sistemine çevrilmesi konusunda anlatılan işlemlerin tam tersi işlem yapılır. Verilen onaltılık sayının her basamağının karşılığı dört basamaklı ikilik sayı şeklinde yazılır. Dört basamaklık gruplar bir araya getirilerek ikilik sayı bulunur.

ÖRNEK : $(3AB)_{16}$ sayısını ikilik sayı sistemine çeviriniz.

ÇÖZÜM: Onaltılık sayının her basamağının karşılığı olan ikilik sayı dört basamak şeklinde yazılır. Dörtlü gruplar bir araya getirilerek ikilik sayı bulunur.

3	A	B
0 0 1 1	1 0 1 0	1 0 1 1

Sonuç=$(0011\ 1010\ 1011)_2$

ÖRNEK : $(AC37F)_{16}$ onaltılık sayısını ikilik sayı sistemine çeviriniz.

CEVAP: Onaltılık sayının her basamağının karşılığı olan ikilik sayı dört basamak şeklinde yazılır. Dörtlü gruplar bir araya getirilerek ikilik sayı bulunur.

A	C	3	7	F
1 0 1 0	1 1 0 0	0 0 1 1	0 1 1 1	1 1 1 1

Sonuç=$(1010\ 1100\ 0011\ 0111\ 1111)_2$

1.3. LOJİK ENTEGRELER

Belirli bir fonksiyonu yerine getirmek için çok sayıda transistör, diyot ve direncin bir araya getirilmesiyle aynı kılıf içerisinde oluşturulan devrelere **entegre (IC)** denir. Entegreler düşük maliyetleri ve çok az yer kaplamaları nedeniyle elektronik endüstrisinde güç yükselteçleri, sayıcılar, aritmetik üniteler, gerilim düzenleyiciler, radyo ve TV devreleri, işlemsel yükselteçler vb. çok geniş kullanım alanlarına sahiptir.

Sayısal devrelerde binary kodlar kullanılmakta ve devrelerin giriş-çıkışları lojik 0 ya da lojik 1 ile ifade edilmektedir. Sayısal bilgileri işleyecek şekilde tasarlanmış entegrelere **lojik entegreler** denir. Sayısal sistemlerin en temel bileşenleri lojik (mantık) kapılardır. Lojik entegreler bünyelerindeki kapı sayısına göre aşağıdaki gibi sınıflandırılır:

SSI [Small Scale Integration (Smol Skeyl Intıgreyşın) Küçük Ölçekli Entegre]: Bünyesinde 1 ile 19 arasında lojik kapı bulundurur. Örneğin 7400 modeli entegre, içerisinde dört adet NAND kapısı bulundurur.

MSI [Medium Scale Integration (Midıım Skeyl Intıgreyşın) Orta Ölçekli Entegre]: Bünyesinde 20 ile 99 arasında lojik kapı bulundurur (flip-floplar, sayıcılar gibi.).

LSI [Large Scale Integration (Larc Skeyl Intıgreyşın) Büyük Ölçekli Entegre]: Bünyesinde 100 ile 9999 arasında lojik kapı bulundurur (4 ve 8 bitlik hafıza elemanları EPROM, ROM gibi.).

VLSI [Very Large Scale Integration (Veri Larc Skeyl Intıgreyşın) Çok Büyük Ölçekli Entegre]: Bünyesinde10000 ile 99999 arasında lojik kapı bulundurur (16-32 bitlik mikroişlemciler, hafıza devreleri, bilgisayar devreleri gibi.).

ULSI [Ultra LargeScale Integration (Ultrı Larc Skeyl Intıgreyşın) Ultra Büyük Ölçekli Entegre]: Bünyesinde 100000 ve fazlası transistör içeren entegrelerdir (çok gelişmiş entegreler gibi.).

Lojik entegreler, adlarını kapı devrelerinin yapımında kullanılan elektronik malzemeden alır. En yaygın kullanılan entegre türleri ise TTL (Transistor-Transistor Logic) ve CMOS'lardır [Complementary Metal Oxide Semiconductor (Kamplımentri Metıl Aksayd Semaykındaktır)].

1.3.1. TTL [Transistor-Transistor Logic (Transistör) Transistör Mantığı] Entegreler

TTL entegre devrelerin tasarımında bipolar transistörler (BJT) kullanılır ve en çok kullanılan lojik entegrelerdir. TTL lojik ailesinde 54 ya da 74 numaralı ön ekler kullanmaktadır. 54 serileri -55 ºC ile 125 ºC sıcaklıkları arasında çalışabilmekte ve askerî amaçlı kullanılmaktadır. 74 serileri ise diğer tüm alanlarda kullanılmakta ve 0 ºC ile 70 ºC aralığında çalışmaktadır. Besleme voltajı 5 V olup voltaj çıkış değeri 2 V ve üstü için lojik 1 çıkışı, voltaj seviyesi 0,0 V-0,8 V aralığında ise lojik 0 çıkışı vermektedir. 0,8 V'tan büyük ve 2 V'tan az herhangi bir voltaj değeri belirsizliktir.

TTL lojik ailesi hız ve güç değerlerine göre yedi alt gruba ayrılmakta ve her grubun kendine özgü bir harf kodu bulunmaktadır. Bu gruplar ve özellikleri Tablo 1.3'te verilmiştir.

Tablo 1.3: TTL Lojik Entegre Grupları ve Özellikleri

TTL Serisi	Harf Kodu	Özelliği	Örnek
Standart TTL	-	En eski, yavaş ve güç kayıpları çok fazla	74XXX
Yüksek güçlü TTL	H	Standart versiyona göre daha yüksek güç	74HXXX
Düşük güçlü (low power) TTL	L	Standart versiyona göre daha az güç kayıpları	74LXXX
Şotki (schotlky) TTL	S	Hızlı fakat güç kayıpları fazla	74SXXX
Düşük güçlü (low power schotlky) TTL	LS	Hızlı ve düşük güç kayıplarına sahip	74LSXXX
Geliştirilmiş (advanced) LS TTL	ALS	Hız-güç kayıpları oranı çok iyi	74ALSXXX
Hızlı (FAST) TTL	F	Hız-güç kayıpları açısından en iyi TTL entegresi	74FXXX

TTL entegreler **AA74YYXX** şeklinde tanımlanır. Burada AA harfleri üretici firmayı gösteren harf kodlarıdır, **YY** harfleri hangi **TTL** serisi olduğunu gösterir. **XXX** ise entegrenin kapı ya da fonksiyonunu gösteren iki veya üç basamaklı bir sayıdır.

1.3.2. CMOS [Complementary Metal Oxide Semiconductor (Tamamlayıcı Metal Oksit Yarı İletken)] Entegreler

CMOS entegre devrelerinin tasarımında alan etkili transistörler (FET ve MOSFET) kullanılır ve TTL entegrelerin daha gelişmiş şeklidir. Ancak çalışma hızları (yayılım hızları) oldukça ya-

vaştır. CMOS entegreler 40 serisi ile belirtilir. 4 'ten sonraki rakamlar entegrenin fonksiyonunu gösterir. 3 V-18 V arasında oldukça geniş bir besleme gerilimi aralığında çalışabilir. Güç harcamaları TTL'ye göre daha azdır.

CMOS lojik ailesi temel olarak metal kapılı CMOS ve silikon kapılı CMOS olmak üzere iki ayrı işlem teknolojisi şeklinde üretilir. Eski metal kapılı teknoloji 4000 serisinden oluşurken yeni silikon kapılı teknolojiler ise 74C, 74HC, 74HCT serisinden oluşur. Bu üç entegre serisi 74 ile başlasa da CMOS entegreleridir. CMOS lojik ailesine ait bütün 74 serisi entegreler, TTL'ler ile aynı bacak ve fonksiyona sahiptir yani 74 serisi TTL ve CMOS entegreler aynı sayıda ve aynı özellikte giriş, çıkış ve besleme gerilimine (Vcc) sahiptir.

1.3.3. TTL ve CMOS Entegrelerin Karşılaştırılması

TTL ile CMOS entegreler arasındaki farklar performans karakteristikleri ile alakalıdır. TTL ve CMOS entegrelerin karşılaştırmalı lojik gerilim değerleri Tablo 1.4'te verilmiştir. Tablo 1.5'te ise TTL ve CMOS entegrelerin performans karşılaştırması verilmiştir.

Tablo 1.4: TTL ve CMOS Entegrelerin Karşılaştırmalı Lojik Gerilim Değerleri

TTL	0 V-0,8 V	Lojik 0	Giriş seviyeleri
	0,8 V-2 V	Belirsiz	
	2 V-5 V	Lojik 1	
CMOS	0 V-1,5 V	Lojik 0	
	1,5 V-3,5 V	Belirsiz	
	3,5 V ve üstü	Lojik 1	
TTL	0 V-0,4 V	Lojik 0	Çıkış seviyeleri
	0,4 V-2,4 V	Belirsiz	
	2,4 V-5 V	Lojik 1	
CMOS	0 V'a çok yakın	Lojik 0	
	Beslemenin %99'u	Lojik 1	

Tablo 1.5: TTL ve CMOS Entegrelerin Performans Karşılaştırması

	TTL	CMOS
Besleme kaynağı	5 V DC	3 V-18 V DC
Gerekli akım	Miliamper	Mikroamper
Giriş empedansı	Düşük	Çok yüksek
Anahtarlama hızı	Hızlı	Yavaş
Çıkış kapasitesi	10	50
Güç harcaması	20 mW	2 mW
Tetikleme palsi	50 MHz	25 MHz
Besleme toleransı	%20	%50

1.3.4. Entegre Parametreleri

Entegrelerin parametreleri aşağıda verilmiştir:

Besleme Voltajı [Power Supply Voltage (Pauır Sıplay Voltıc)]: Entegrenin besleme gerilimini belirler.

Yayılım Gecikmesi [Propagation Delay (Prapıgeyşın Dîley)]: Bir lojik devrenin girişine verilen bilgiye göre çıkışın değişim hızını nano saniye cinsinden gösterir.

Güç Harcaması [Power Dissipation (Pauır Disipeyşın)]: Miliwatt cinsinden devrenin harcadığı güç miktarını gösterir. Devrenin çalışma hızıyla doğru orantılıdır.

Çıkış Kapasitesi [Fan Out (Faen Aut)]: Devrenin çıkışına bağlanacak maksimum yük miktarını belirler.

Gürültü Miktarı [Noise İmmunity (Noy İmyuniti)]: Gürültü miktarı devrenin çıkışındaki bilgilerin hata oranını belirler. Çıkışta 1 değerli sinyalin 0 değerli sinyal gibi veya bunun tersi gibi görünmesi durumudur. Gürültü miktarı (milivolt cinsinden) ne kadar az olursa çıkış bilgileri de o kadar hatasız olur.

Pals Frekansı [Clockfrequency (Klakfrikwınsi)]: Dijital devrenin girişine uygulanacak tetikleme palsinin MHz cinsinden frekansını belirler.

İdeal bir entegrenin özellikleri şöyle sıralanabilir:

- Hızlı çalışmalıdır.
- Güç harcaması minimum olmalıdır.
- Ekonomik olmalıdır.
- Isı değişimlerinden etkilenmemelidir.
- Gürültü başlığı iyi olmalıdır.
- Hata miktarı **0 (sıfır)** olmalıdır.

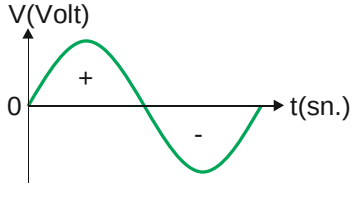
1.3.5. Low Aktif, High Aktif Giriş ve Çıkışlar

Entegrelere uygulanacak elektriksel parametreler aşağıda verilmiştir:

Lojik (Mantık): Öne sürülen düşüncelere göre karar vermeye **lojik** denir.

Analog Sinyal: Maksimum ve minimum seviyeler arasında farklı değerler alabilen kesintisiz, sürekli bir sinyaldir. Belli bir zaman dilimindeki örnekleme sayısı sınırsızdır.

Dijital (Sayısal) Sinyal: Yalnızca iki değer alabilen (var-yok, kapalı-açık, evet-hayır low- high vb.) sinyallerdir yani lojik 1 ve lojik 0 değerleri vardır. Lojik 0, 0 V veya GND'yi ifade ederken lojik 1 +5 V'u ifade eder. Sayısal sinyaller sürekli bir sinyal değildir yani belli bir zaman dilimindeki sayısı sınırlıdır. Görsel 1.2a'da analog sinyal şekli görülmektedir. Görsel 1.2b'de dijital sinyal şekli görülmektedir.



Görsel 1.2a: Analog sinyal



Görsel 1.2b: Dijital sinyal

Sayısal işaretin aldığı değerler zıplayarak yani adım adım değişir. Lojik sistemlerde pozitif ve negatif mantık olmak üzere iki durum mevcuttur. Sayısal işarette 0'dan 1'e ani değişim pozitif yönde ise **pozitif mantık**, ani değişim negatif yönde ise **negatif mantık** olarak tanımlanır.

Lojik **1** değeri doğruyu, olumlu durumu, aktif çalışmayı temsil etmek; lojik **0** değeri ise yanlışı, olumsuz durumu, aktif olmama durumunu temsil etmek için kullanılıyorsa bu mantık pozitif mantık olarak isimlendirilir. **Pozitif mantık kullanan sistemde high seviye 1, low seviye 0'ı temsil eder.**

Lojik 0 değeri doğruyu, olumlu durumu, aktif çalışmayı temsil etmek; lojik **1** değeri ise yanlışı, olumsuz durumu, aktif olmama durumunu temsil etmek için kullanılıyorsa bu mantık negatif mantık olarak isimlendirilir. **Negatif mantık kullanan sistemde high seviye 0, low seviye 1'i temsil eder.**

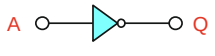
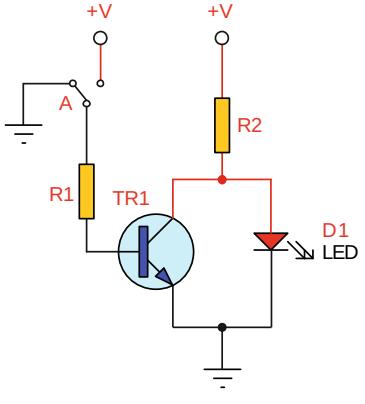
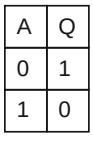
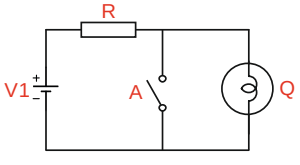
1.4. LOJİK KAPILAR

Anahtarlama devrelerinin matematikteki mantıksal işlemleri yapabilecek şekilde birbirine bağlanmasıyla oluşturulan devrelere **lojik kapılar** denir. Dijital elektronikte kullanılan toplam yedi adet lojik kapı vardır.

Lojik kapılar dijital elektroniğin temelini oluşturur. İkilik sayı sisteminde işlem yapar. Her bir kapının belli bir işlevi bulunmaktadır. Dijital elektronikte **0 yokluğu** (anahtar açık durumunu, şase) **1 ise varlığı** (+5 V, anahtar kapalı durumunu) ifade eder. **Lojik 0** değeri analog olarak 0 V ile 1,5 V gerilim aralığını, **lojik 1** değeri ise analog olarak 2 V ile 5 V gerilim aralığını ifade eder. Lojik kapıların birleşmesi ile lojik entegreler meydana gelir.

1.4.1. DEĞİL Kapısı (NOT GATE-INVERTER)

DEĞİL kapısı değilleme işlemi yapar. Girişinden uygulanan sinyali ters çevirir. Giriş sinyali 1 ise çıkış sinyali 0'dır, giriş sinyali 0 ise çıkış sinyali 1'dir. Tek girişi vardır. Görsel 1.3'te DEĞİL kapısının özellikleri görülmektedir.

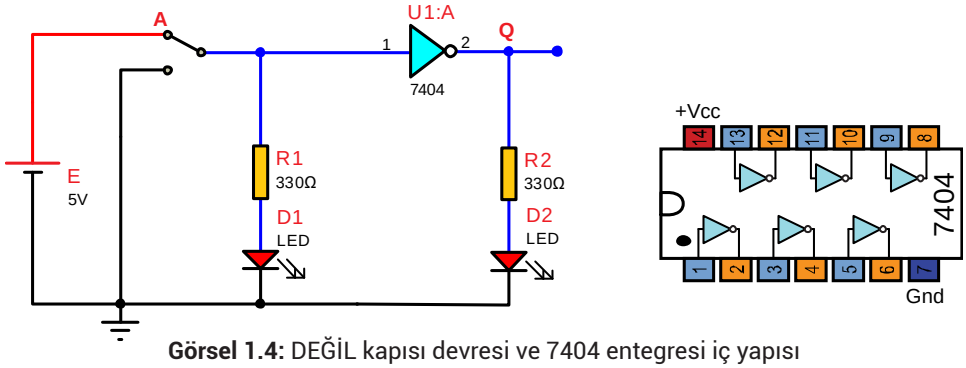
$Q=\overline{A}$		
Boolean denklemi	DEĞİL (NOT) kapısı sembolü	
		
Doğruluk tablosu	Elektriksel eş değeri	Transistör eş değeri

Görsel 1.3: DEĞİL kapısı

Uygulama Adı: DEĞİL Kapısı (NOT GATE) Uygulaması No.: 1

Amaç: DEĞİL kapısı (NOT GATE) uygulaması yapmak.

Görsel 1.4'te 7404 entegresi ile yapılmış DEĞİL kapısı uygulama devresi ve 7404 entegresi iç yapısı görülmektedir. 7404 entegresinde altı adet NOT kapısı bulunmaktadır.



Görsel 1.4: DEĞİL kapısı devresi ve 7404 entegresi iç yapısı

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 1.4'teki devreyi kurunuz.
4. Besleme kaynağının (+) ucunu 7404 entegresinin 14 numaralı ayağına, (-) ucunu ise 7 numaralı ayağına bağlayınız. Devreye enerji veriniz.
5. Giriş anahtarlarının konumunu Tablo 1.6'ya uygun şekilde ayarlayarak Q çıkış ifadesinin durumlarını tabloya yazınız.

SIRA SİZDE

1. Görsel 1.4'teki değil kapısının yerine transistör eşleniğini kullanarak devreyi kurunuz.

Değerlendirme

Tablo 1.6: DEĞİL Kapısı Doğruluk Tablosu

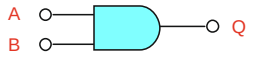
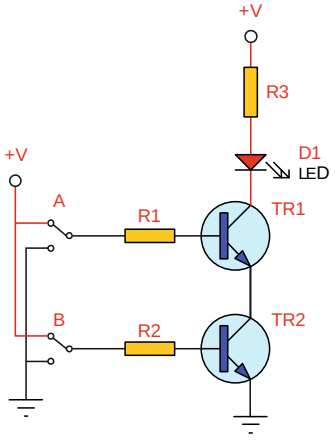
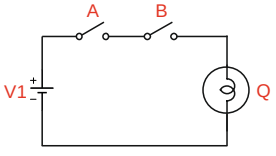
A	Q
0	
1	

Sorular

1. DEĞİL kapısının matematiksel ifadesini yazınız.
2. DEĞİL kapısının eş değer elektrik devresini çiziniz.
3. İki adet DEĞİL kapısı arka arkaya bağlanırsa çıkış ifadesi ne olur? Açıklayınız.

1.4.2. VE Kapısı (AND GATE)

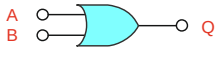
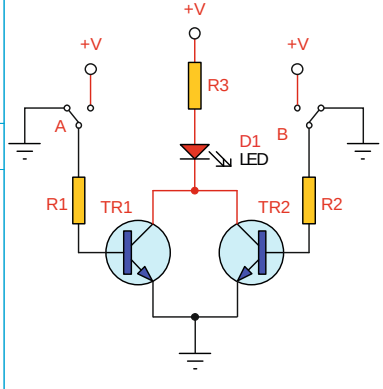
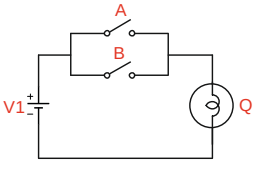
VE kapısı matematikteki çarpma işlemi gibi çalışır. Çarpma işleminde 0 yutan elemandır. En az iki girişi olmalıdır. Girişlerinden herhangi biri 0 ise çıkış sinyali 0'dır. Giriş sinyallerinin hepsi 1 ise çıkış sinyali 1'dir. Görsel 1.5'te VE kapısının özellikleri görülmektedir.

<table>
<tr><td>Q=AB</td><td> 
</td><td rowspan="3"> 
</td></tr>
<tr><td>Boolean denklemi</td><td>VE (AND) kapısı sembolü</td></tr>
<tr><td><table>
<tr><th>A</th><th>B</th><th>Q</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table></td><td> 
</td></tr>
<tr><td>Doğruluk tablosu</td><td>Elektriksel eş değeri</td><td>Transistör eş değeri</td></tr>
</table>

Görsel 1.5: VE (AND) kapısı

1.4.3. VEYA Kapısı (OR GATE)

VEYA kapısı matematikte toplama işlemi gibi çalışır. Toplama işleminde 0 etkisiz elemandır. En az iki girişi olmalıdır. Girişlerinden herhangi biri 1 ise çıkış sinyali 1'dir. Giriş sinyallerinin hepsi 0 ise çıkış sinali 0'dır. Görsel 1.6'da VEYA kapısının özellikleri görülmektedir.

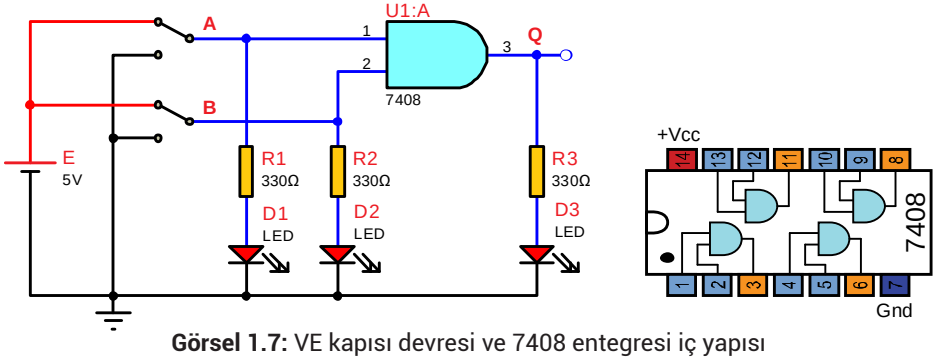
<table>
<tr><td>Q=A+B</td><td> 
</td><td rowspan="3"> 
</td></tr>
<tr><td>Boolean denklemi</td><td>VEYA (OR) kapısı sembolü</td></tr>
<tr><td><table>
<tr><th>A</th><th>B</th><th>Q</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table></td><td> 
</td></tr>
<tr><td>Doğruluk tablosu</td><td>Elektriksel eş değeri</td><td>Transistör eş değeri</td></tr>
</table>

Görsel 1.6: VEYA (OR) kapısı

Uygulama Adı: VE Kapısı (AND GATE) Uygulaması — No.: 2

Amaç: VE Kapısı (AND GATE) uygulaması yapmak.

Görsel 1.7'de 7408 entegresi ile yapılmış iki girişli, bir çıkışlı VE kapısı uygulama devresi ve 7408 entegresi iç yapısı görülmektedir. Bu entegrenin içerisinde birbirinden bağımsız dört adet VE kapısı vardır.



Görsel 1.7: VE kapısı devresi ve 7408 entegresi iç yapısı

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 1.7'deki devreyi kurunuz.
4. Besleme kaynağının (+) ucunu 7408 entegresinin 14 numaralı ayağına, (-) ucunu ise 7 numaralı ayağına bağlayınız. Devreye enerji veriniz.
5. Giriş A, B anahtarlarının konumlarını Tablo 1.7'ye uygun şekilde ayarlayarak Q çıkış ifadesinin durumlarını tabloya yazınız.

SIRA SİZDE

1. Entegrenin 4, 5 ve 6 numaralı ayaklarını kullanarak işlem basamaklarını tekrar ediniz.

Değerlendirme

Tablo 1.7: VE Kapısı Doğruluk Tablosu

A	B	Q
0	0	
0	1	
1	0	
1	1	

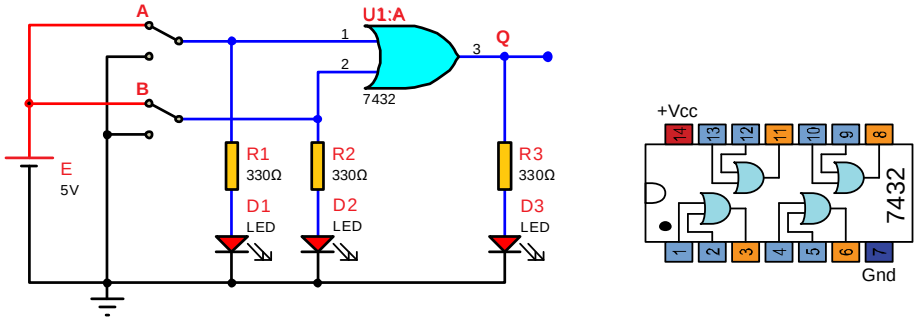
Sorular

1. VE kapısının matematiksel ifadesini yazınız.
2. VE kapısının eş değer elektrik devresini çiziniz.
3. Çıkış ne zaman lojik 1 olur? Açıklayınız.

Uygulama Adı: VEYA Kapısı (OR GATE) Uygulaması No.: 3

Amaç: VEYA kapısı (OR GATE) uygulaması yapmak.

Görsel 1.8'de 7432 entegresi ile yapılmış iki girişli, bir çıkışlı VEYA kapısı uygulama devresi ve 7432 entegresi iç yapısı görülmektedir.



Görsel 1.8: VEYA kapısı devresi ve 7432 entegresi iç yapısı

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 1.8'deki devreyi kurunuz.
4. Besleme kaynağının (+) ucunu 7432 entegresinin 14 numaralı ayağına, (-) ucunu ise 7 numaralı ayağına bağlayınız. Devreye enerji veriniz.
5. Giriş A, B anahtarlarının konumlarını Tablo 1.8'e uygun şekilde ayarlayarak Q çıkış ifadesinin durumlarını tabloya yazınız.

SIRA SİZDE

1. Görsel 1.8'deki VEYA kapısının yerine transistör eşleniğini kullanarak devreyi kurunuz.

Değerlendirme

Tablo 1.8: VEYA Kapısı Doğruluk Tablosu

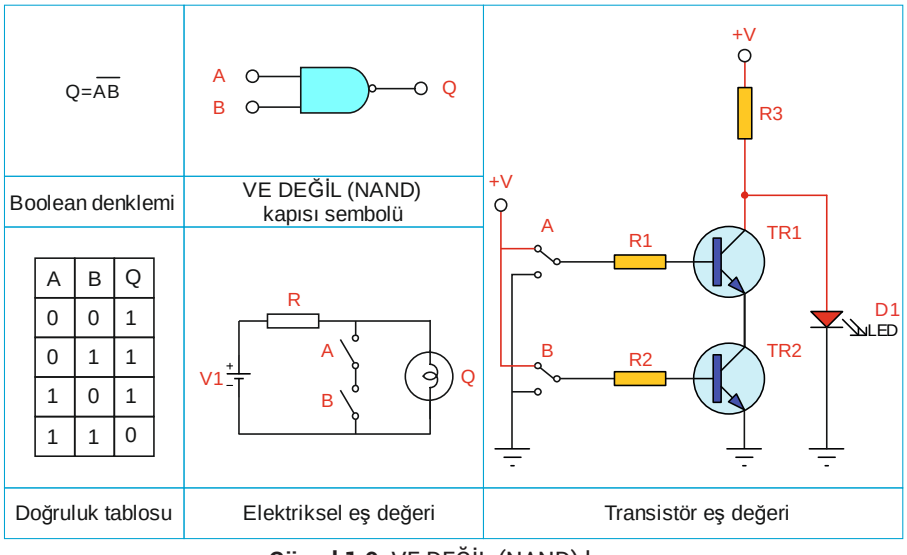
A	B	Q
0	0	
0	1	
1	0	
1	1	

Sorular

1. VEYA kapısının matematiksel ifadesini yazınız.
2. VEYA kapısının eş değer elektrik devresini çiziniz.
3. Çıkış ne zaman lojik 0 olur? Açıklayınız.

1.4.4. VE DEĞİL Kapısı (NAND GATE)

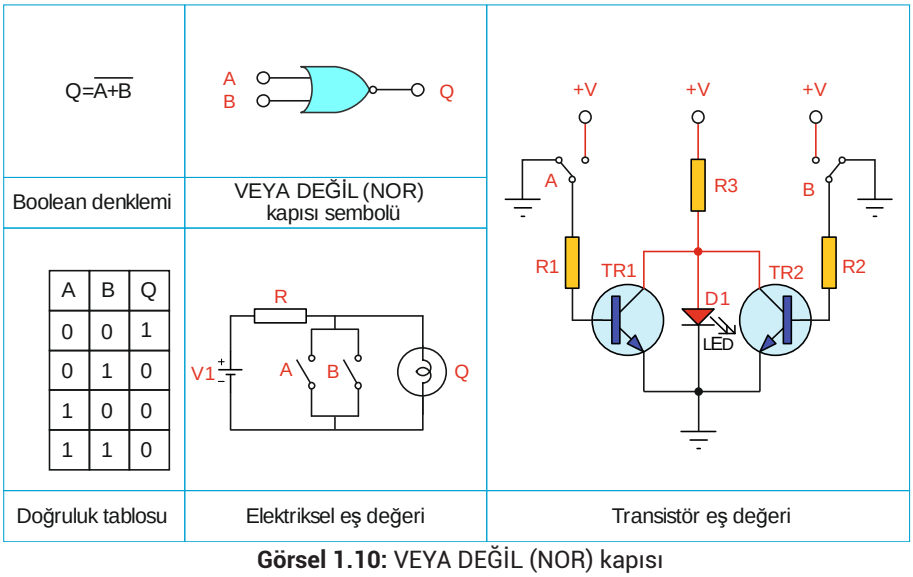
VE DEĞİL kapısı VE kapısının çıkışına DEĞİL kapısı eklenerek elde edilmiştir. VE DEĞİL kapısı önce çarpma işlemi yapar, sonra çıkan sonucun değilini alır. En az iki girişi vardır. Çarpma işleminde 0 yutan elemandır. Girişlerinden herhangi biri 0 ise çarpma sonucu 0 olur ve değili alındığı için çıkış sinyali 1'dir. Giriş sinyallerinin hepsi 1 ise çıkış sinyali 0'dır. Görsel 1.9'da NAND kapısının özellikleri görülmektedir.



Görsel 1.9: VE DEĞİL (NAND) kapısı

1.4.5. VEYA DEĞİL Kapısı (NOR GATE)

VEYA DEĞİL kapısı VEYA kapısının çıkışına DEĞİL kapısı eklenerek elde edilir. VEYA DEĞİL kapısı önce toplama işlemi yapar, sonra çıkan sonucun değilini alır. En az iki girişi vardır. Toplama işleminde 0 etkisiz elemandır. Girişlerinden herhangi biri 1 ise toplama sonucu 1 olur ve değili alındığı için çıkış sinyali 0'dır. Giriş sinyallerinin hepsi 0 ise çıkış sinyali 1 olur. Görsel 1.10'da VEYA DEĞİL kapısının özellikleri görülmektedir.

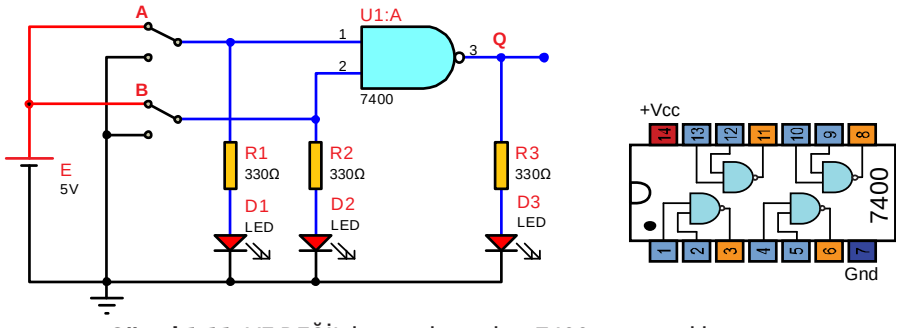


Görsel 1.10: VEYA DEĞİL (NOR) kapısı

Uygulama Adı: VE DEĞİL Kapısı (NAND GATE) Uygulaması No.: 4

Amaç: VE DEĞİL kapısı (NAND GATE) uygulaması yapmak.

Görsel 1.11'de 7400 entegresi ile yapılmış iki girişli bir çıkışlı VE DEĞİL kapısı uygulama devresi ve 7400 entegresi iç yapısı görülmektedir. VE DEĞİL kapısının girişleri birleştirilerek DEĞİL kapısı olarak da kullanılabilir.



Görsel 1.11: VE DEĞİL kapısı devresi ve 7400 entegresi iç yapısı

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 1.11'deki devreyi kurunuz.
4. Besleme kaynağının (+) ucunu 7400 entegresinin 14 numaralı ayağına, (-) ucunu ise 7 numaralı ayağına bağlayınız. Devreye enerji veriniz.
5. Giriş A, B anahtarlarının konumlarını Tablo 1.9'a uygun şekilde ayarlayarak Q çıkış fadesinin durumlarını tabloya yazınız.

SIRA SİZDE

1. Entegrenin 4, 5 ve 6 numaralı ayaklarını kullanarak işlem basamaklarını tekrar ediniz.

Değerlendirme

Tablo 1.9: VE DEĞİL Kapısı Doğruluk Tablosu

A	B	Q
0	0	
0	1	
1	0	
1	1	

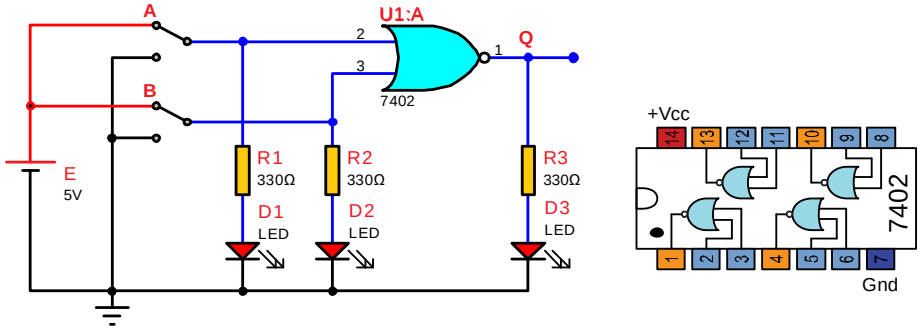
Sorular

1. VE DEĞİL kapısının matematiksel ifadesini yazınız.
2. VE DEĞİL kapısının eş değer elektrik devresini çiziniz.
3. İki girişli VE DEĞİL kapılarını kullanarak üç girişli VE DEĞİL kapısı şeklini çiziniz.

Uygulama Adı: VEYA DEĞİL Kapısı (NOR GATE) Uygulaması No.: 5

Amaç: VEYA DEĞİL Kapısı (NOR GATE) Uygulaması yapmak.

Görsel 1.12'de 7402 entegresi ile yapılmış iki girişli bir çıkışlı VEYA DEĞİL kapısı uygulama devresi ve 7402 entegresi iç yapısı görülmektedir.



Görsel 1.12: VEYA DEĞİL kapısı devresi ve 7402 entegresi iç yapısı

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.

2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.

3. Görsel 1.12'deki devreyi kurunuz.

4. Besleme kaynağının (+) ucunu 7402 entegresinin 14 numaralı ayağına, (-) ucunu ise 7 numaralı ayağına bağlayınız. Devreye enerji veriniz

5. Giriş A, B anahtarlarının konumlarını Tablo 1.10'a uygun şekilde ayarlayarak Q çıkış ifadesinin durumlarını tabloya yazınız.

SIRA SİZDE

1. Entegrenin 4, 5 ve 6 numaralı ayaklarını kullanarak işlem basamaklarını tekrar ediniz.

Değerlendirme

Tablo 1.10: VEYA DEĞİL Kapısı Doğruluk Tablosu

A	B	Q
0	0	
0	1	
1	0	
1	1	

Sorular

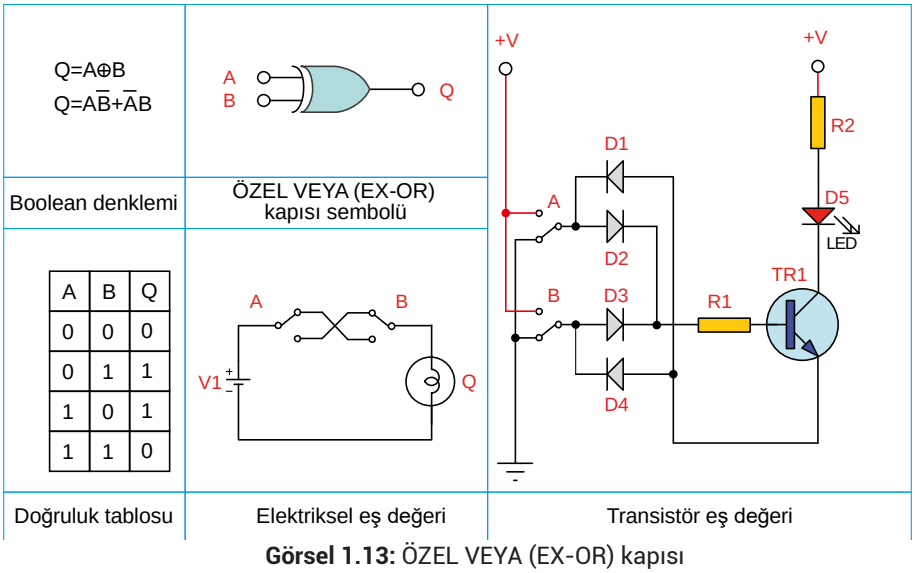
1.VE DEĞİL kapısının matematiksel ifadesini yazınız

2. VE DEĞİL kapısının eş değer elektrik devresini çiziniz.

3. Çıkış ne zaman lojik 1 olur? Açıklayınız.

1.4.6. ÖZEL VEYA Kapısı (EX-OR GATE)

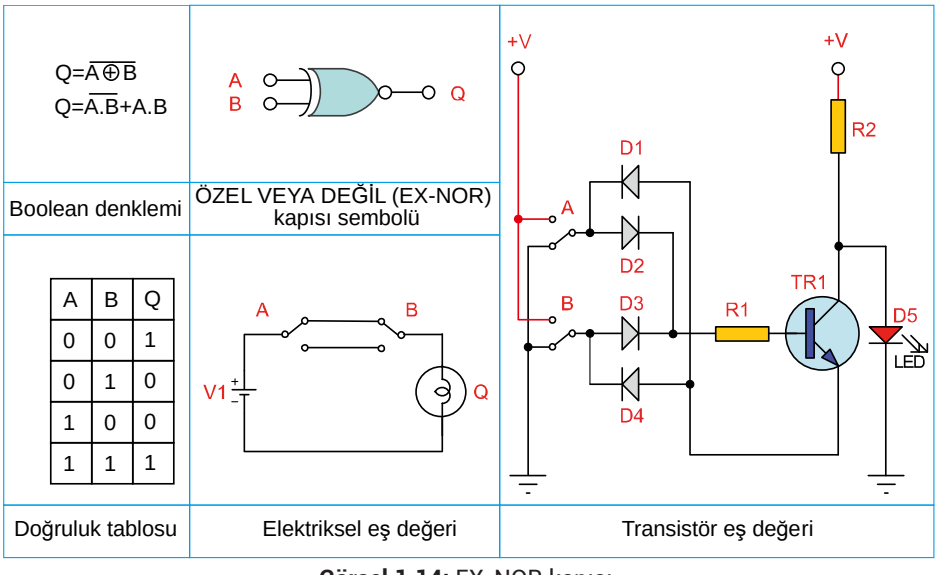
ÖZEL VEYA kapısı karşılaştırma yapabilen iki girişli lojik kapıdır. Giriş sinyalleri aynı ise (0-0 ve 1-1) çıkış sinyali 0, giriş sinyalleri farklı ise (0-1 ve 1-0) çıkış sinyali 1'dir. Özel toplama kapısı da denmektedir. Görsel 1.13'te ÖZEL VEYA kapısının özellikleri görülmektedir.



Görsel 1.13: ÖZEL VEYA (EX-OR) kapısı

1.4.7. ÖZEL VEYA DEĞİL Kapısı (EX-NOR GATE)

EX-NOR kapısı EX-OR kapısının çıkışına değil kapısı eklenerek elde edilmiştir. EX-NOR kapısı önce özel toplama işlemi yapar, sonra çıkan sonucun değilini alır. Giriş sinyalleri aynı ise (0-0 ve 1-1) çıkış sinyali 1, giriş sinyalleri farklı ise (0-1 ve 1-0) çıkış sinyali 0'dır. Görsel 1.14'te EX-NOR kapısının özellikleri görülmektedir.

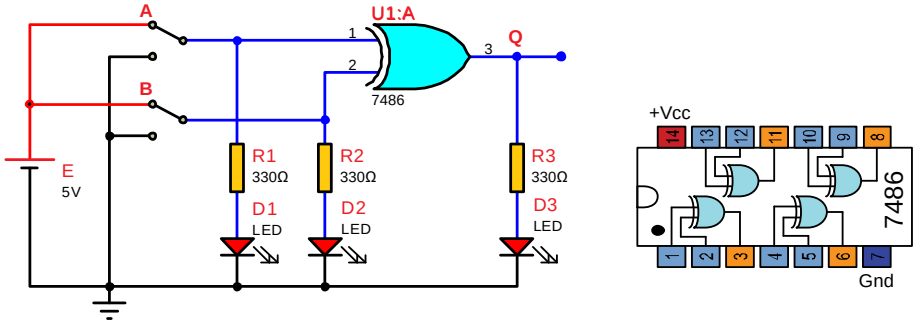


Görsel 1.14: EX-NOR kapısı

Uygulama Adı: Özel Veya Kapısı (Exor Gate) Uygulaması No.: 6

Amaç: ÖZEL VEYA kapısı (EXOR GATE) uygulaması yapmak.

Görsel 1.15'te 7486 entegresi ile yapılmış iki girişli bir çıkışlı ÖZEL VEYA kapısı uygulama devresi ve 7486 entegresi iç yapısı görülmektedir.



Görsel 1.15: ÖZEL VEYA kapısı devresi ve 7486 entegresi iç yapısı

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 1.15'teki devreyi kurunuz.
4. Besleme kaynağının (+) ucunu 7486 entegresinin 14 numaralı ayağına, (-) ucunu ise 7 numaralı ayağına bağlayınız. Devreye enerji veriniz.
5. Giriş A, B anahtarlarının konumlarını Tablo 1.11'e uygun şekilde ayarlayarak Q çıkış ifadesinin durumlarını tabloya yazınız.

SIRA SİZDE

1. Entegrenin 4, 5 ve 6 numaralı ayaklarını kullanarak işlem basamaklarını tekrar ediniz.

Değerlendirme

Tablo 1.11: ÖZEL VEYA Kapısı Doğruluk Tablosu

A	B	Q
0	0	
0	1	
1	0	
1	1	

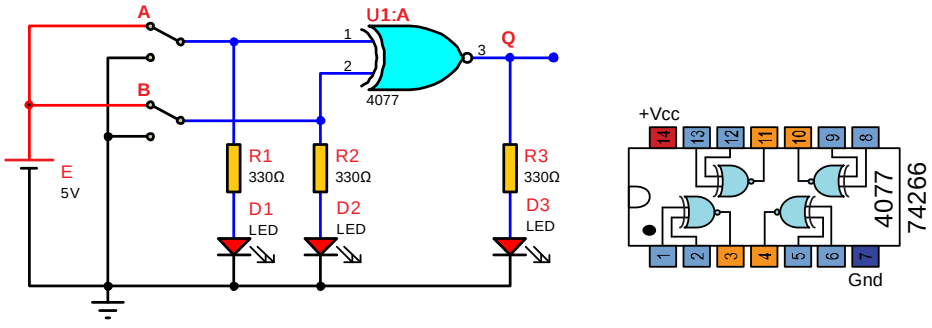
Sorular

1. ÖZEL VEYA kapısının matematiksel ifadesini yazınız.
2. ÖZEL VEYA kapısının eş değer elektrik devresini çiziniz. Diğer kapıları kullanarak ÖZEL VEYA kapısını elde ediniz.

Uygulama Adı: Isı (Exnor Gate) Uygulaması No.: 7

Amaç: ÖZEL VEYA DEĞİL kapısı (EXNOR) uygulaması yapmak.

Görsel 1.16'da 4077 (eş değeri 74266) entegresi ile yapılmış iki girişli, bir çıkışlı ÖZEL VEYA DEĞİL kapısı uygulama devresi ve 4077 entegresi iç yapısı görülmektedir.



Görsel 1.16: ÖZEL VEYA DEĞİL kapısı devresi ve 4077 entegresi iç yapısı

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 1.16'daki devreyi kurunuz.
4. Besleme kaynağının (+) ucunu 4077 entegresinin 14 numaralı ayağına, (-) ucunu ise 7 numaralı ayağına bağlayınız. Devreye enerji veriniz.
5. Giriş A, B anahtarlarının konumlarını Tablo 1.12'ye uygun şekilde ayarlayarak Q çıkış ifadesinin durumlarını tabloya yazınız.

SIRA SİZDE

1. Entegrenin 4, 5, ve 6 numaralı ayaklarını kullanarak işlem basamaklarını tekrar ediniz.

Değerlendirme

Tablo 1.12: ÖZEL VEYA DEĞİL Kapısı Doğruluk Tablosu

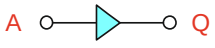
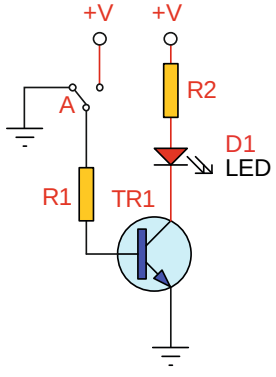
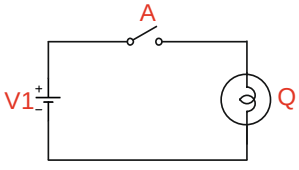
A	B	Q
0	0	
0	1	
1	0	
1	1	

Sorular

1. ÖZEL VEYA DEĞİL kapısının matematiksel ifadesini yazınız.
2. ÖZEL VEYA DEĞİL kapısının eş değer elektrik devresini çiziniz.
3. 40xx serisi CMOS entegreler ile 74xx serisi TTL entegreler arasındaki farklar nelerdir? Araştırınız.

1.4.8. TAMPON Kapısı (BUFFER GATE)

TAMPON kapısı girişinden uygulanan sinyali aynen çıkışına aktarır ve bir girişi ile bir çıkışı vardır. Giriş sinyali 1 ise çıkış sinyali 1, giriş sinyali 0 ise çıkış sinyali 0'dır. Görsel 1.17'de TAMPON kapısının özellikleri görülmektedir.

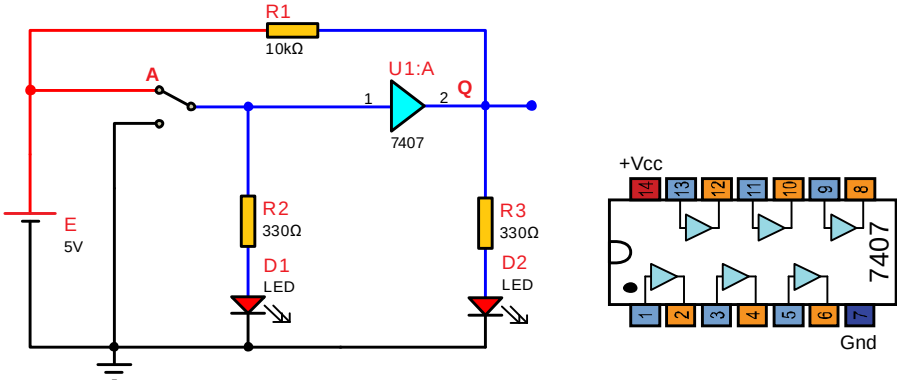
<table>
<tr><td>Q=A</td><td> 
</td><td rowspan="2"> 
</td></tr>
<tr><td>Boolean denklemi</td><td>TAMPON (BUFFER) sembolü</td></tr>
<tr><td><table><tr><td>A</td><td>Q</td></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table></td><td> 
</td><td></td></tr>
<tr><td>Doğruluk tablosu</td><td>Elektriksel eş değeri</td><td>Transistör eş değeri</td></tr>
</table>

Görsel 1.17: TAMPON (BUFFER) kapısı

Uygulama Adı: TAMPON Kapısı (BUFFER GATE) Uygulaması No.: 8

Amaç: TAMPON kapısı (BUFFER GATE) uygulaması yapmak.

Görsel 1.18'de TAMPON kapısı devresi ve 7407 entegresi iç yapısı görülmektedir. Kapı çıkışı **opencollector** (opınkollektör) olduğu için R1 direnci kullanılmıştır.



Görsel 1.18: TAMPON kapısı devresi ve 7407 entegresi iç yapısı

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 1.18'deki devreyi kurunuz.
4. Besleme kaynağının (+) ucunu 7404 entegresinin 14 numaralı ayağına, (-) ucunu ise 7 numaralı ayağına bağlayınız. Devreye enerji veriniz.
5. Giriş anahtarının konumunu Tablo 1.13'e uygun şekilde ayarlayarak Q çıkış ifadesinin durumlarını tabloya yazınız.

SIRA SİZDE

1. Entegrenin 3, 4 ve 5, 6 numaralı ayaklarını kullanarak işlem basamaklarını tekrar ediniz.

Değerlendirme

Tablo 1.13: DEĞİL Kapısı Doğruluk Tablosu

A	Q
0	
1	

Sorular

1. DEĞİL kapısının matematiksel ifadesini yazınız.
2. DEĞİL kapısının eş değer elektrik devresini çiziniz.
3. İki adet DEĞİL kapısı arka arkaya bağlanırsa çıkış ifadesi ne olur? Açıklayınız.

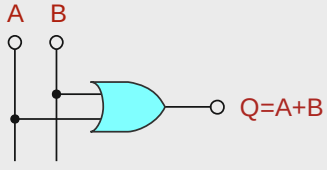
1.5. LOJİK KAPI UYGULAMALARI

Boolean ifadeler lojik kapılar ile gösterilebildiği gibi lojik devreler de boolean ifadeler ile gösterilebilir.

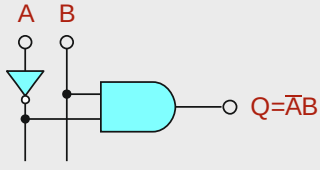
1.5.1. Lojik İfadelerinin Lojik Devrelerle Gösterimi

Boolean ifadelerinin lojik kapılarla çizim örnekleri aşağıda görülmektedir.

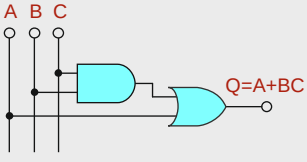
Örnek 1: Q=A+B



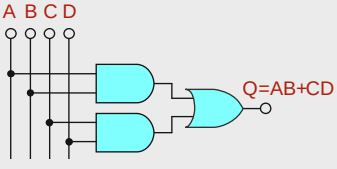
Örnek 2: Q=$\overline{A}$B



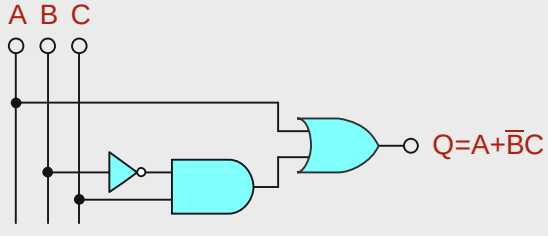
Örnek 3: Q=A+BC



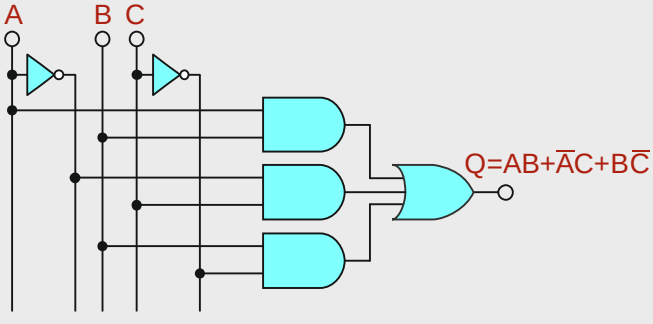
Örnek 4: Q=AB+CD



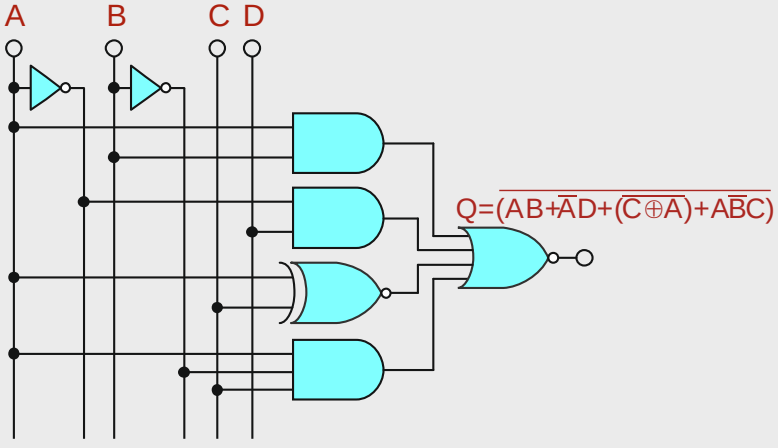
Örnek 5: Q=A+$\overline{B}$C



Örnek 6: Q=AB+$\overline{A}$C+B$\overline{C}$



Örnek 7: $Q=\overline{(AB+\overline{A}D+\overline{(C\oplus A)}+A\overline{B}C)}$



SIRA SİZDE

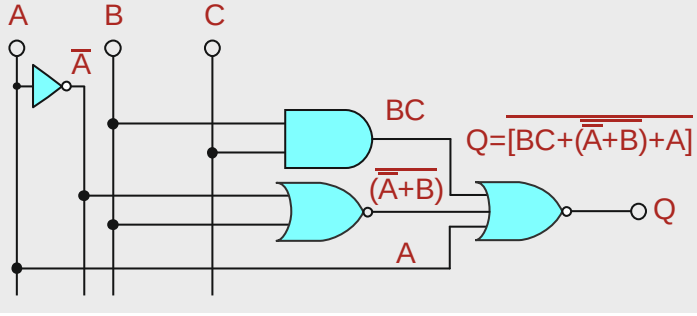
1. Aşağıdaki boolean ifadelerini lojik kapılarla çiziniz.

Soru 1: $Q=AB+CD$
Soru 2: $Q=(\overline{A}+B)(C+\overline{D})$
Soru 3: $Q=\overline{A}BC$
Soru 4: $Q=A\,\overline{B}\,\overline{C}\,(A+D)$
Soru 5: $Q=(\overline{\overline{AB}+CD})$
Soru 6: $Q=(\overline{\overline{C}+\overline{A}\,\overline{B}})$
Soru 7: $Q=A+(\overline{\overline{A}B+\overline{B}\,\overline{C}})$
Soru 8: $Q=(A\oplus B\oplus C)D$
Soru 9: $Q=A\,\overline{B}\,\overline{C}+\overline{A}\,B\,\overline{C}$
Soru 10: $Q=(\overline{AB+BC+CD})$

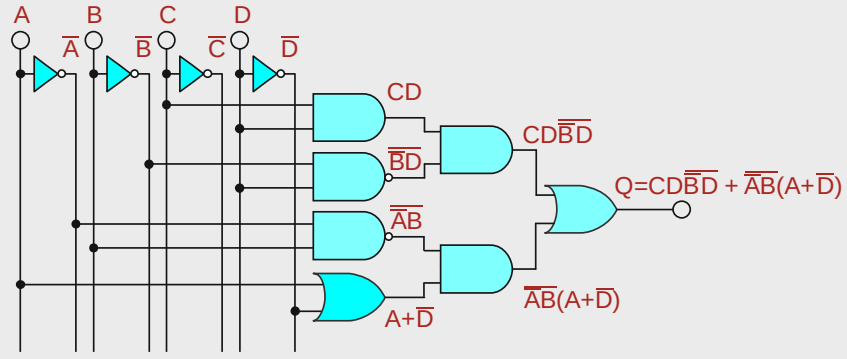
1.5.2. Lojik Devrelerin Lojik İfadelerle Gösterimi

Lojik kapı çizimlerinin üzerine boolean formülleri yazılarak tasarımı yapılan devrelerin boolean karşılıkları elde edilir.

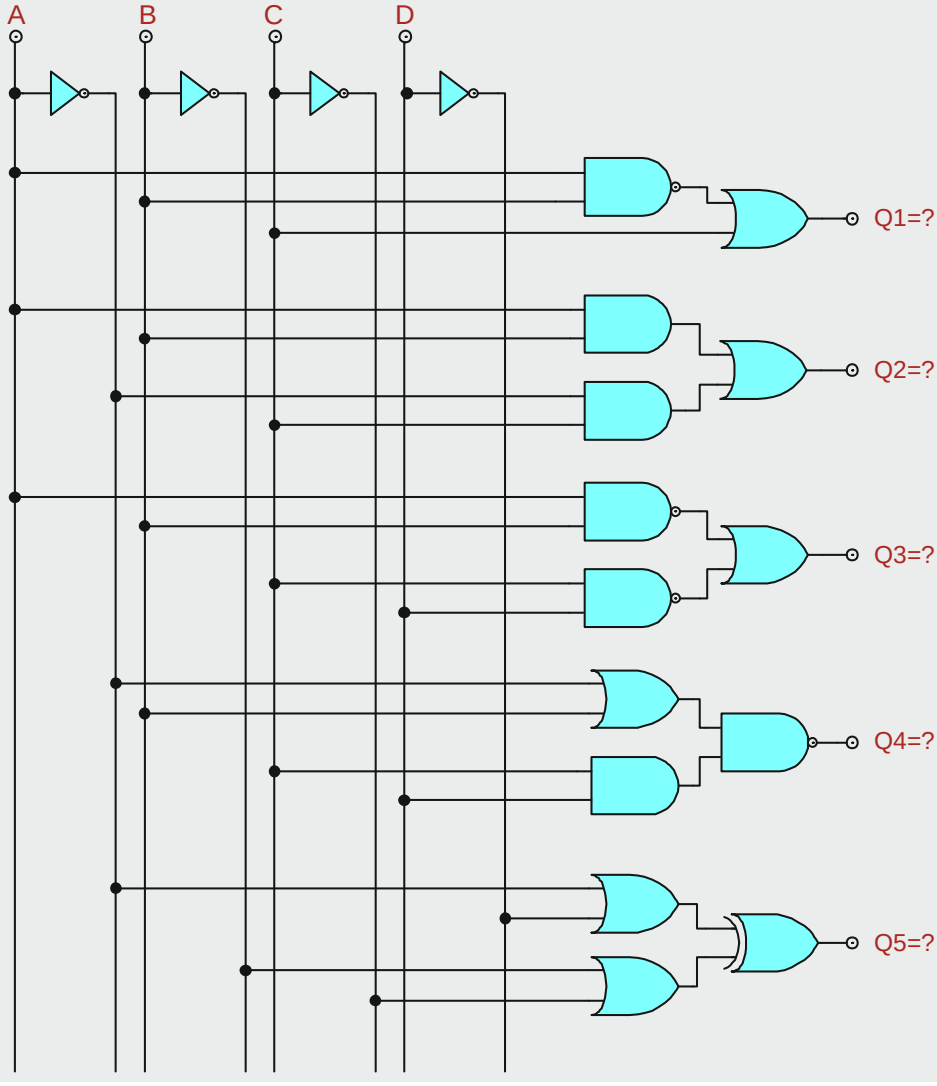
Örnek 1: Q=?



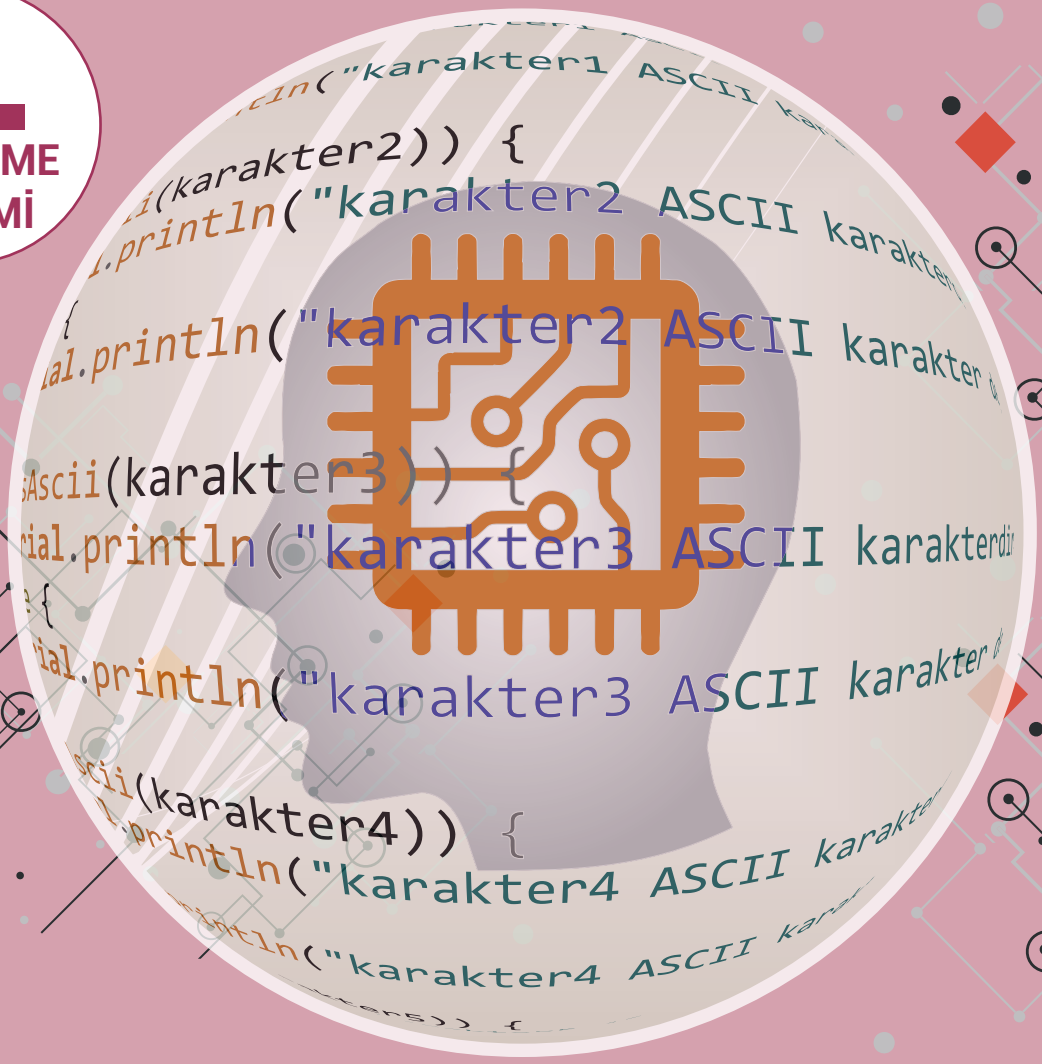
Örnek 2: Q=?



Aşağıda verilen lojik devrelerin boolean çıkış denklemlerini yazınız.



2.
ÖĞRENME BİRİMİ



MİKRODENETLEYİCİ VE PROGRAMLAMA

NELER ÖĞRENECEKSİNİZ ?

- Mikroişlemciler ve Mikrodenetleyiciler
- Mikrodenetleyici Editör Programı
- Mikrodenetleyiciye Program Yükleme
- Algoritma Hazırlama
- Dijital Giriş Çıkış İşlemleri
- Seri Port Analog Giriş Çıkış İşlemleri

TEMEL KAVRAMLAR

mikrodenetleyici
algoritma
fonksiyon
koşul
döngü
değişken
EEPROM
işlem
mantık
dijital
analog
PWM
kesme

2. 1. MİKROİŞLEMCİLER VE MİKRODENETLEYİCİLER

Mikroişlemci, işlemci ana işlem biriminin [CPU Central Process Unit (Sentrıl Proses Yunıt)] fonksiyonlarını tek bir yarı iletken tüm devrede [IC Integrated Circuit (İntıgreytıd Sırkıt)] birleştiren, programlanabilir bir sayısal elektronik bileşendir. Kullanıcı ya da programcı tarafından yazılan programları meydana getiren komutları veya bilgileri yorumlamak ve yerine getirmek için gerekli olan tüm mantıksal devreleri kapsar. Bu devreler genelde transistörlerden meydana gelmektedir.

2.1.2. Mikrodenetleyici

Bir mikroişlemcili sistemi meydana getiren temel bileşenlerden mikroişlemci, bellek ve G/Ç birimlerinin, bazı özelliklerinin kırpılarak (azaltılarak) tek bir entegre içerisinde üretilmiş biçimine **mikrodenetleyici (microcontroller)** denir. Denetim teknolojisi gerektiren uygulamalarda kullanılmak üzere tasarlanmış olan mikrodenetleyiciler, mikroişlemcilere göre çok daha basit ve ucuzdur. Endüstrinin her kolunda kullanılan mikrodenetleyiciler; otomobil, kamera, cep telefonu, fotokopi ve çamaşır makineleri, televizyon, oyuncak vb. cihazlarda sıklıkla kullanılmaktadır.

2.1.3. Mikroişlemci ve Mikrodenetleyici Arasındaki Farklar

Bir mikroişlemcinin çalışabilmesi için en az üç ayrı ek devre gerekir. Bunlar yazılan programın tutulacağı kalıcı bellek birimi, değişken verilerin saklanacağı geçici bellek birimi ve dış dünyadan veri alışverişinin yapılmasını sağlayan giriş/çıkış birimleridir. Bunlar ve daha fazlası bir mikroişlemcili sistemde ayrı ayrı devreler şeklinde yerini alır. Bu sebeple mikroişlemcili sistemlere çok entegreli sistemler de denir. Mikroişlemci ve diğer gerekli olabilecek çeşitli devreler, bir entegre devrede birleştirilmiş ve mikrodenetleyiciler üretilmiştir.

Bilgisayar gibi mikroişlemcili sistemlere verilen bir örnekte, bir bilgisayarın bir çamaşır makinesinde veya cep telefonunda kullanılması elbette mümkün olmayacaktır. Bilgisayar aynı anda milyonlarca işi yapabildiğinden ve çok yer kapladığından böyle yerlerde kullanılması mantıklı değildir ve maliyeti çok olur. Bu sebeple sistemi meydana getiren elemanların birçok özelliğinden feragat edilerek ve elemanlar bir entegrede birleştirilerek mikroişlemcilerin yeni türevleri oluşturulmuştur.

2.1.4. Mikrodenetleyici Çeşitleri

Mikrodenetleyicilerin üretildikleri firmalara göre pek çok çeşidi vardır. Piyasada Atmel, Texas Instruments, ST Microelectronics, Microchip firmalarının mikrodenetleyicilerini kullanan pek çok geliştirme kartı mevcuttur. Atmel mikrodenetleyicileri kullanılan Arduino geliştirme kartları, Arduino topluluğu tarafından donanım tasarımları ve geliştirme ortamı kodları açık kaynak kodlu olarak herkesin kullanımına sunulmuştur. Bu donanım tasarımlarını kullanarak herkes kendine özgü geliştirme kartları tasarlayıp kullanabilir. Bu kitapta Atmega328p mikrodenetleyicisini kullanan Arduino Uno ve Nano modelleri üzerinden mikrodenetleyici programlama teknikleri gösterilmiştir.

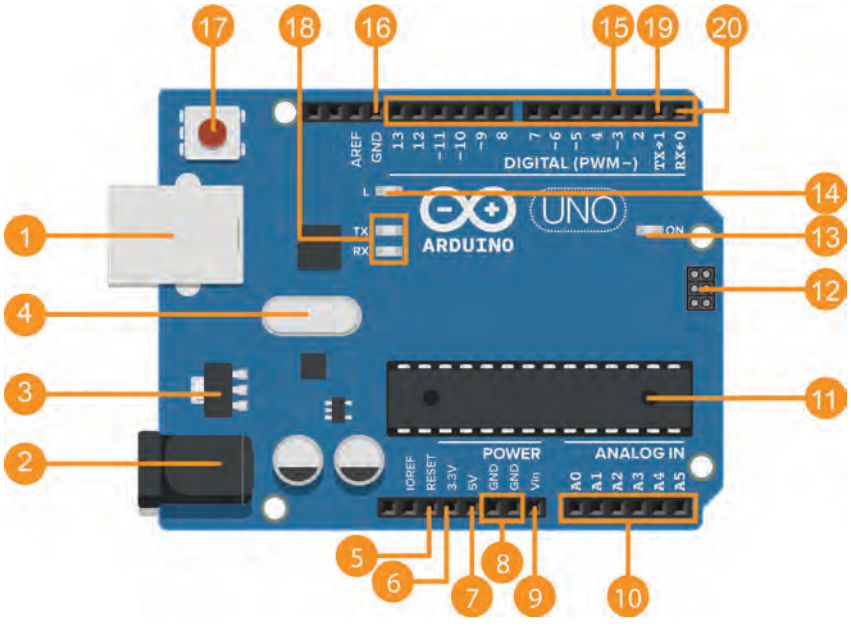
2.2. MİKRODENETLEYİCİ KARTININ DONANIM YAPISI VE ÖZELLİKLERİ

Arduino mikrodenetleyicisinde kullanılan işlemci Microchip firması bünyesindeki Atmele ait ATmega328P'dir. Geliştirme kartı Arduino geliştirici topluluğuna aittir. Mikrodenetleyicinin temel özellikleri aşağıda maddeler hâlinde verilmiştir:

1. 7 V-12 V besleme voltajı
2. 5 V çalışma voltajı

3. Pin başına 20 mA akım (en fazla 40 mA)
4. 32 kB program belleği (0,5 kB'ı bootloader (önyükleyici) tarafından kullanılan Arduino kodunun yüklendiği kalıcı hafıza)
5. 2 kB RAM bellek (geçici hafıza)
6. 1 kB EEPROM bellek (kalıcı hafıza)
7. 16 MHz çalışma frekansı
8. 2 x 8 bit 1 x 16 bit toplam 3 adet timer
9. 1 x I2C, 2 x SPI, 1 UART

Görsel 2.1'de Arduino Uno kartı ve donanımsal yapısı görülmektedir.



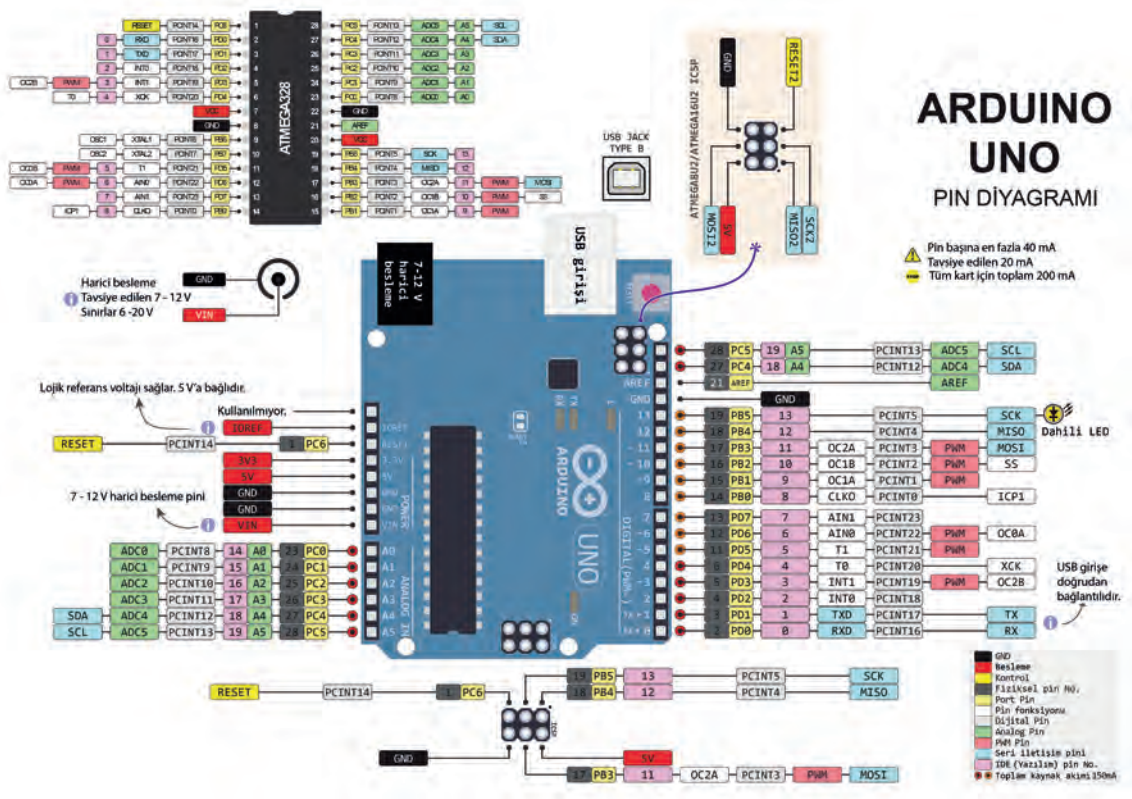
Görsel 2.1: Arduino Uno kartı ve donanımsal yapısı

1. USB girişi: Arduino ile bilgisayar arasındaki iletişimi sağlar. Arduino 5 V besleme gerilimini USB üzerinden alır.
2. Haricî besleme: Arduino'nun çalışması için gereken güç, DC adaptör veya pil ile bu soket üzerinden sağlanır (7 V-12 V).
3. Voltaj regülatörü: Haricî beslemeden gelen 7 V-12 V aralığındaki gerilimi 5 V'a dönüştürür.
4. Kristal osilatör: Arduino işlemcisinin çalışma frekansını üretir.
5. Reset: Çalışan kodu baştan başlatır.
6. 3,3 V çıkış sağlar (50 mA).
7. 5 V çıkış sağlar.
8. GND (ground-toprak) 0 V besleme pinidir.
9. Vin: Pin şeklindeki 7 V-12 V haricî besleme girişidir.
10. Analog In: Altı adet analog pin bulunmaktadır. A0...A5 olarak isimlendirilir. Dijital giriş çıkış olarak da kullanılır. ADC 10 bittir.
11. ATmega328P mikrodenetleyicisidir.

12. ICSP: Bu pinler SPI kütüphanesiyle SPI haberleşmeyi sağlar.
13. Power LED: Arduino'ya besleme verildiğinde yanar.
14. Dâhilî LED: Arduino'nun 13 No.lu pinine bağlı LED'dir.
15. On dört adet dijital giriş / çıkış pinleridir. Bazı pinlerin özel fonksiyonları vardır.
16. 0 (RX) ve 1 (TX) pinleri seri data almak [receive (risiv) RX] ve yaymak [transmit (tranzmit) TX] için kullanılır. 2 ve 3 numaralı pinler haricî kesme [interrupt (intıapt)] pinleridir. 3, 5, 6, 9, 10 ve 11 numaralı PWM pinleri analogWrite() fonksiyonuyla 8-bit PWM sinyali sağlar.
17. AREF: Analog girişler için referans voltajıdır. analogReference() fonksiyonuyla kullanılır.

Arduino Uno Pin Diyagramı

Arduino'da pinlerin birçok işlevi vardır. Görsel 2.2'de uygulamalarda kullanılan özellikler için ayrıntılı pin diyagramı verilmiştir.



Görsel 2.2: Arduino Uno pin diyagramı

Bellekler

Avr tabanlı Arduino kartlarında kullanılan mikrodenetleyicide üç tip bellek vardır:

- Flash bellek (program alanı), Arduino programının yüklendiği yerdir.
- SRAM (statik rastgele erişimli bellek), taslak çalıştığında geçici değişkenleri oluşturduğu ve yönettiği yerdir.
- EEPROM, bilgileri depolamak için (sabit disk gibi) kullanılan bellek alanıdır.

Flash bellek ve EEPROM bellek kalıcıdır (Bilgi, güç kapatıldıktan sonra da saklanır devam eder.). SRAM geçicidir. Güç kapatıldığında bilgi kaybolur.

Ardunio Uno'da bulunan ATmega328P yongası aşağıdaki bellek miktarlarına sahiptir:

- Flash 32 kB (0,5 kB'ı bootloader tarafından kullanılır.)
- SRAM 2 kB
- EEPROM 1 kB

Ardunio Mega2560'taki ATmega2560, daha büyük bellek alanına sahiptir:

- Flash 256 kB (8 kB'ı bootloader tarafından kullanılır.)
- SRAM 8 kB
- EEPROM 4 kB

Uno'da çok fazla SRAM bulunmadığı için programda çok sayıda dizi bulunduğu zaman hafıza kolayca tükenir. SRAM biterse program başarıyla yükleniyor gibi görünse de program çalışmayabilir. Bu durumda program çalışmayacak veya garip bir şekilde çalışacaktır. SRAM bitmesinden kaynaklı bir problem olup olmadığını anlayabilmek için diziler programdan kaldırılır. Daha sonra başarılı bir şekilde çalışırsa muhtemelen SRAM bitiyor demektir. Bu sorunu çözmek için aşağıdaki işlemler yapılabilir:

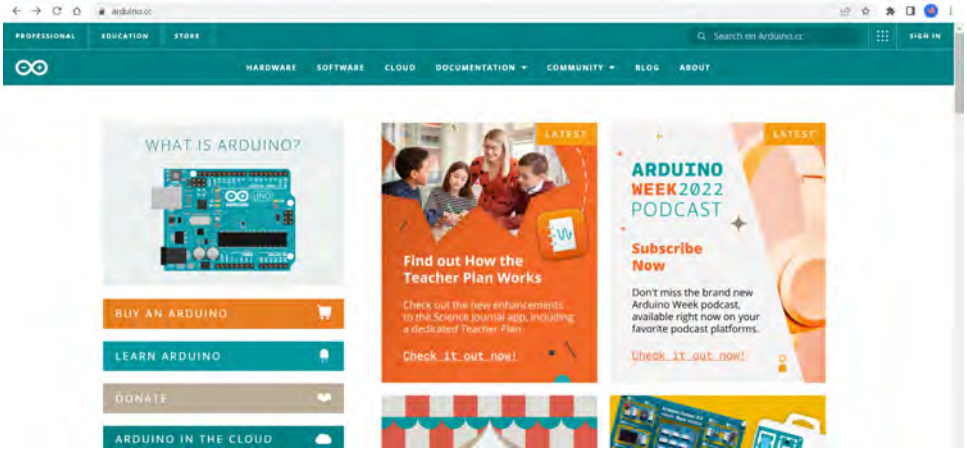
Yazılan program, bilgisayarda çalışan bir programla beraber çalışıyorsa veriler veya hesaplamalar bilgisayardaki programa kaydırılarak Arduino üzerindeki yük azaltılabilir.

Değerleri depolamak için gereken en küçük veri türü kullanılmalıdır. Örneğin int veri tipi iki byte yer kaplarken byte veri tipi için yalnızca bir byte'lık yer kullanılır (Değeri ne kadar küçük olsa da en küçük bilgi bile hafızada bir byte yer kaplamaktadır.).

Taslakta çalışırken dizileri veya verileri değiştirmek gerekmiyorsa bunlar SRAM yerine flash (program) bellekte saklanabilir. Bunu yapmak için PROGMEM anahtar sözcüğü kullanılır. Serial.print() fonksiyonuyla kullanılan F() makrosu da benzer işlev görmektedir.

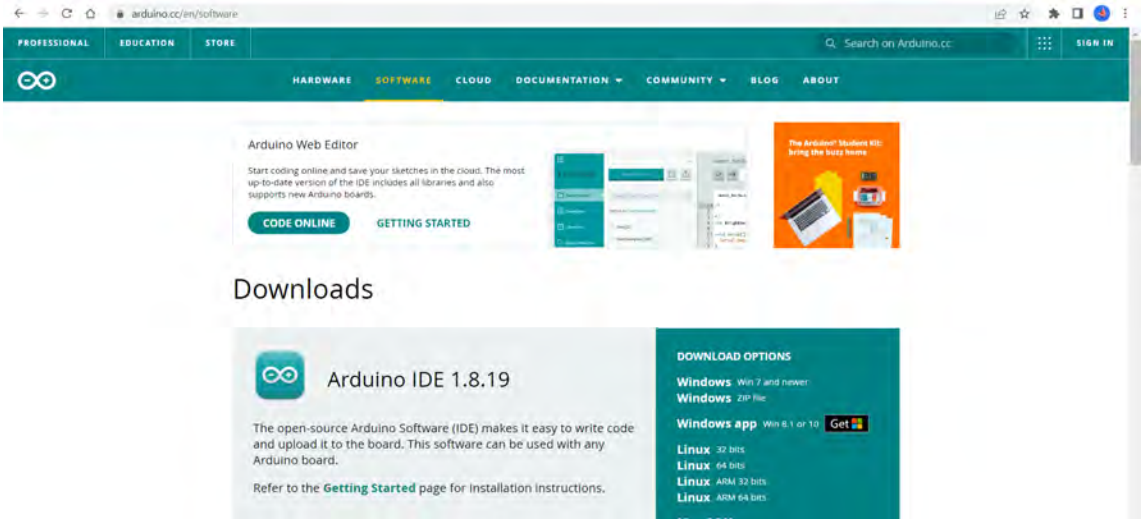
2.3. MİKRODENETLEYİCİ EDİTÖR PROGRAMI

Arduino yazılımını indirmek için www.arduino.cc adresinden software sekmesi tıklanır (Görsel 2.3).



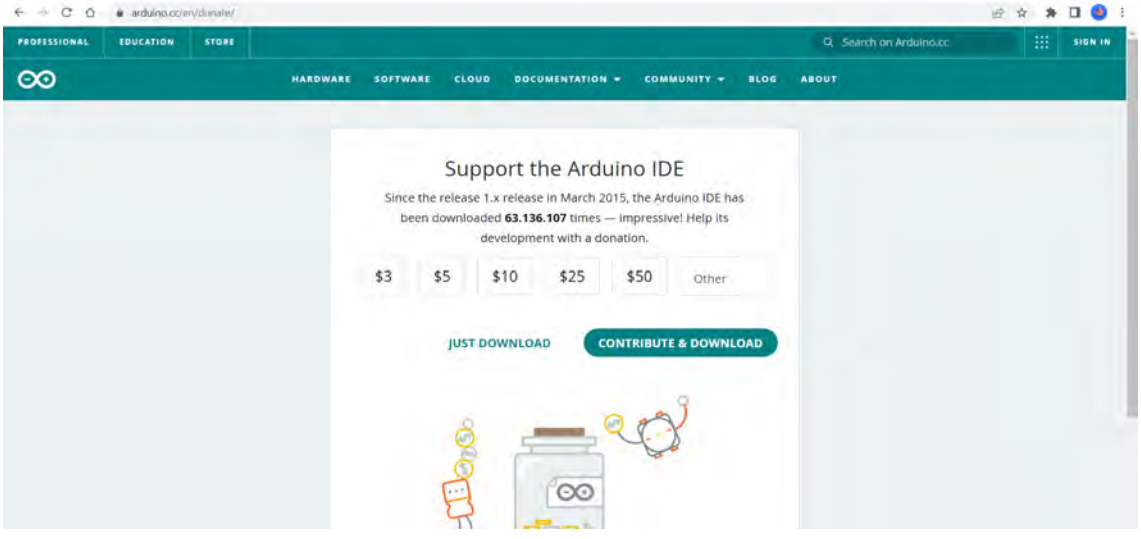
Görsel 2.3: Arduino resmî sitesi

İndirme sayfasında Arduino IDE'nin yayımlanmış son sürümü bulunmaktadır. Sağ taraftaki indirme seçeneklerinden işletim sistemine uygun seçenek tıklanır (Görsel 2.4).



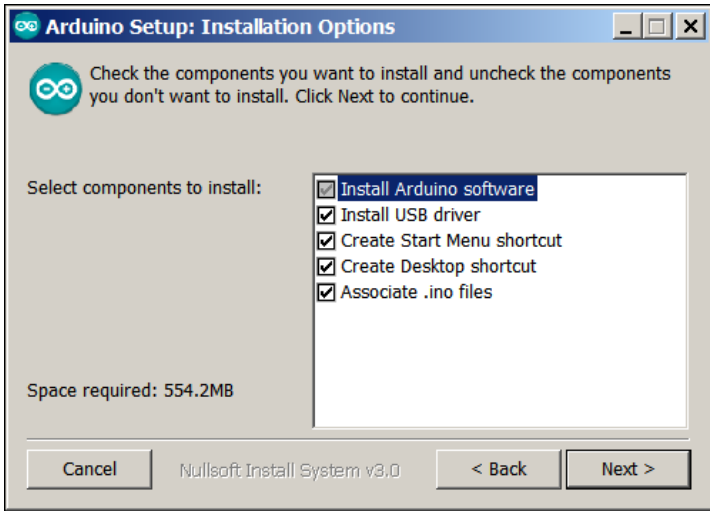
Görsel 2.4: Arduino IDE indirme sayfası

Just download düğmesine tıklanarak Arduino yazılımı bilgisayara indirilir (Görsel 2.5).



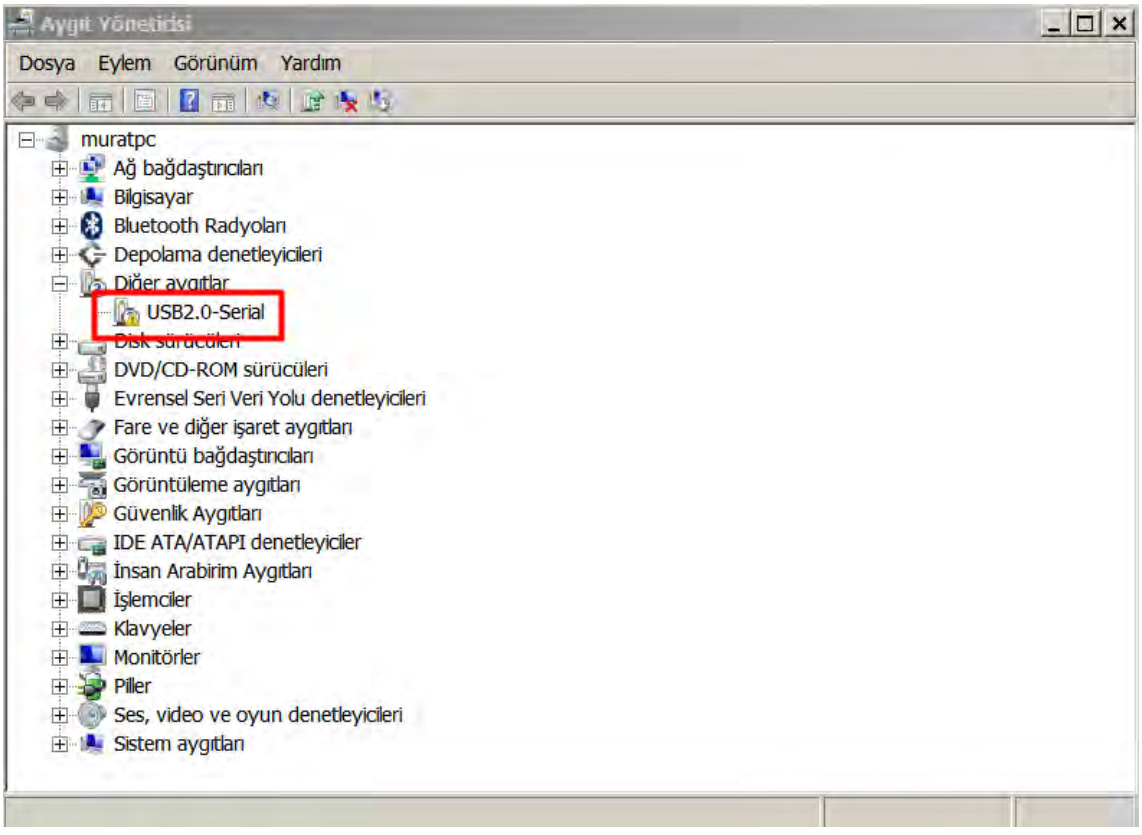
Görsel 2.5: Arduino IDE indirme sayfası

İndirilen kurulum dosyasıyla Arduino yazılımı bilgisayara kurulur (Görsel 2.6).

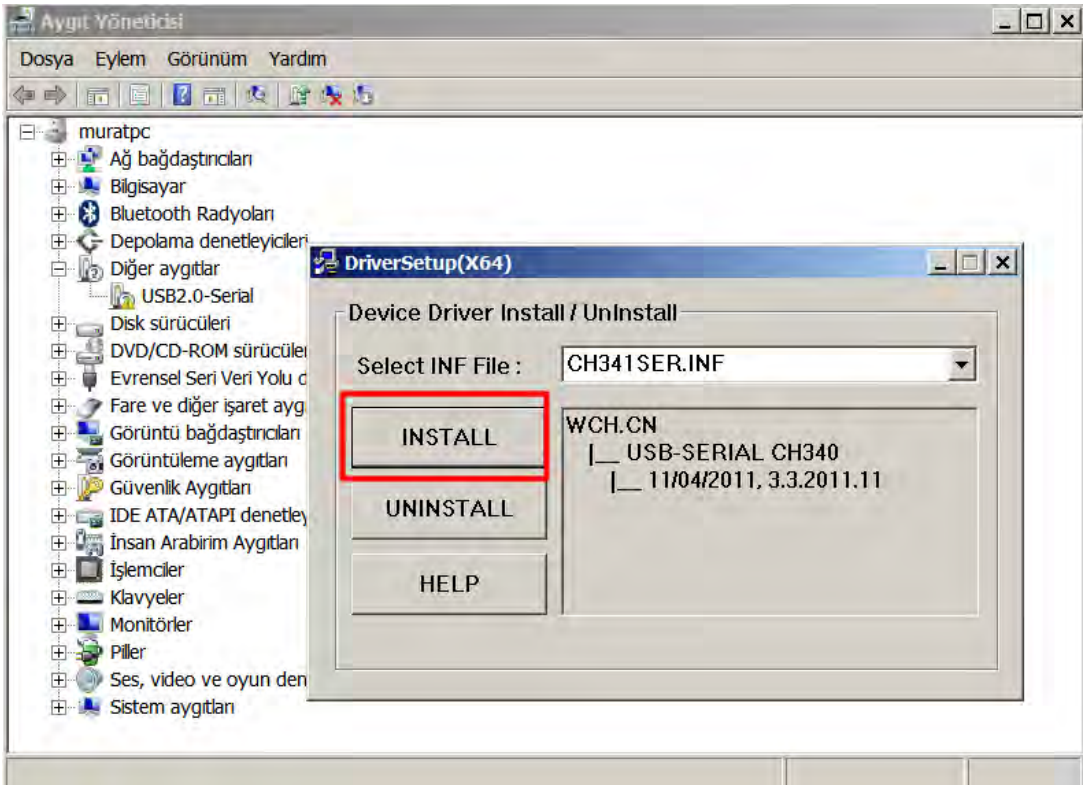


Görsel 2.6: Arduino IDE kurulumu

Arduino üzerinde USB bağlantıyı seri UART bağlantıya dönüştüren ATmega16u2 isimli çip bulunur. Ancak bazı kartlarda CH340 isimli bir çip mevcuttur. Arduino kartı bilgisayara bağlıyken aygıt yöneticisinde Görsel 2.7'deki sürücü yükleme hatası varsa Görsel 2.8'deki CH340.exe isimli sürücü dosyası yüklenir.

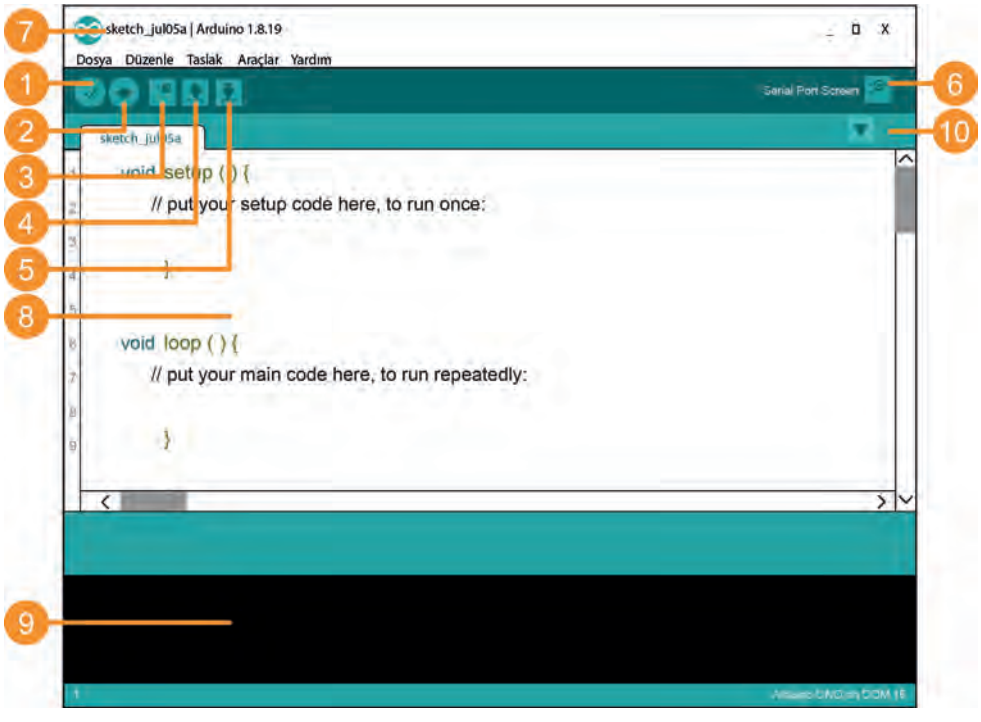


Görsel 2.7: Aygıt yöneticisi sürücü yükleme hatası



Görsel 2.8: CH340 sürücü dosyası yükleme

Sade bir arayüze sahip olan Arduino yazılımındaki butonlar ve görevleri Görsel 2.9'da verilmiştir.



Görsel 2.9: Arduino IDE üzerindeki butonlar

Sade bir arayüze sahip olan Arduino yazılımındaki butonların görevleri aşağıda verilmiştir:

1. Derleme düğmesi yazılan kodu derler ve söz dizimi hatalarını bulur. Mantıksal hataları bulmaz
2. Yükleme düğmesi programı Arduino kartına yükler. Bu sırada RX ve TX pinleri meşguldür.
3. Yeni düğmesi yeni kod penceresi açar.
4. Aç düğmesi bilgisayarda bulunan .ino uzantılı Arduino dosyasını açar.
5. Kaydet düğmesi yazılan programı kaydeder.
6. Seri Monitör düğmesi seri iletişimi görüntüler.
7. Çalışılan programın dosya adı yer alır.
8. Programın yazıldığı alandır.
9. Mesaj alanı derleme ve yükleme butonlarına basıldığında bir hata varsa hatayı açıklamasıyla birlikte kırmızı olarak gösterir.
10. Sekme kullanım seçenekleri yer alır.

Arduino IDE Menüler

Dosya Menüsü: Dosya işlemlerinin yapıldığı komutlar bulunur.

Yeni: Yeni sketch (taslak) oluşturur.

Aç: Bilgisayarda bulunan .ino uzantılı Arduino dosyasını açar.

Sonuncuyu Aç: Son kullanılan dosyaları açmak için kullanılır.

Taslak Defteri: Programların kaydedildiği klasördeki program listesini görüntüler.

Örnekler: Arduino IDE'le gelen örnek kodlar bulunur.

Kapat: Editörü kapatır.

Kaydet: Üzerinde çalışılan program dosyasını kaydeder.

Farklı Kaydet: Üzerinde çalışılan program dosyasını farklı isimle veya farklı adrese kaydeder.

Sayfa Ayarları: Yazıcıdan çıktı almak için sayfa ayarlarını düzenler.

Yazdır: Yazıcıdan programın çıktısını alır.

Tercihler: Arduino programıyla ilgili ayarlar yapılır. Program dosyalarının kaydedileceği klasör, font büyüklüğü, satır numaraları gösterme, kod katlama vb. seçenekler bulunur.

Çıkış: Programı kapatır.

Düzenle Menüsü: İşlemi geri alma, ileri alma, kesme, kopyalama, yapıştırma, seçme, düzenleme işlemleri yapılır.

Geri Al (Undo): Kod yazarken yapılan son işlem geri alınır.

Tekrarla: Geri alınan işlem tekrar gerçekleştirilir (İleri alınır.) .

Kes: Seçilen metni keserek panoya kopyalar.

Kopyala: Seçilen metni panoya kopyalar.

Forum İçin Kopyalama: Yazılan kodları forumlarda paylaşmak üzere panoya kopyalar.

HTML Olarak Kopyalama: Yazılan kodları web sayfalarına entegre etmek için HTML olarak panoya kopyalar (Renkli çıktı almak için kullanılabilir.).

Yapıştır: Panodaki metni imlecin bulunduğu yere yapıştırır.

Tümünü Seç: Tüm kodu seçer.

Satıra Git: İmleci, yazılan satır numarasında gösterir.

Yorum Yap/Yorumu Kaldır: Bir satırı otomatik olarak yorum hâline getirir (Seçilen alanın başına // ekler.). Tekrarında eski hâline getirir.

Girintiyi Artır: Satır başına boşluk ekler.

Girintiyi Azalt: Satır başındaki boşluğu azaltır.

Yazı Tipi Boyutunu Küçült/Büyüt: Yazı tipi boyutunu küçültmek büyütmek için kullanılır. Ctrl + mouse tekerleği kullanılarak da aynı işlem yapılır.

Bul: Kodların içinde aradığımız metni bulur.

Sonrakini Bul: Aranan metnin bir sonraki örneğini bulur.

Öncekini Bul: Aranan metnin bir önceki örneğini bulur.

Taslak Menüsü: Kod derleme, yükleme, kütüphane ekleme gibi işlemlerin yapıldığı menüdür.

Doğrula/Derle: Kodu derler.

Yükle: Kodları derler ve karta yükler.

Programlayıcıyı Kullanarak Yükle: Farklı programlayıcı kullanarak kod yüklemesi yapmak için kullanılır.

Derlenmiş Binaryi Çıkar: Hex dosyasını .ino dosyasıyla aynı klasöre çıkarır.

Çalışma Klasörünü Göster: Derlenen kodların bulunduğu klasörü ve içeriğini gösterir.

Library Ekle: Program yazarken kullanacağımız kütüphane dosyasını kodların başına ekler.

Dosya Ekle: Başka bir konumdaki program dosyasını sketch içine ekler. Eklenen dosya yeni bir sekmede görünür. Eklenen dosya mevcut .ino dosyasıyla aynı klasöre kopyalanır.

Araçlar Menüsü: Kullanılan kartın ve portun seçimi, seri monitörü görüntüleme, oluşturulan programı sıkıştırılmış dosya şeklinde kaydetme gibi işlemlerin yapıldığı menüdür.

Otomatik Biçimlendir: Koddaki girinti ve boşlukları standart yazım şeklinde ayarlar.

Taslağı Arşivle: Yazılan programı, sıkıştırılmış dosya (.zip formatı) olarak kaydeder.

Karakter Kodlamasını Düzelt & Tekrar Yükle: Programda karakter kodlamalarıyla ilgili bir problem varsa bu hataları düzeltir. Web siteleri ve forumlardan alınan kodlarda hata veren durumlarda kullanılır.

Kütüphane Yöneticisi: Kurulu olan veya internetten indirilebilen kütüphaneleri yönetir.

Seri Port Ekranı: Seri iletişim monitörünü açar ve buradan gelen verileri görüntüler.

Seri Çizici: Arduino Serial Monitor'den gelen seri verileri grafik olarak gösterir.

Kart: Kart seçimini yapar.

Port: Bilgisayarla kart arasında haberleşme amacıyla oluşturulan COM portunun seçimi yapılır.

Kart Bilgisi Al: Arduino kartının seri numarasını gösterir.

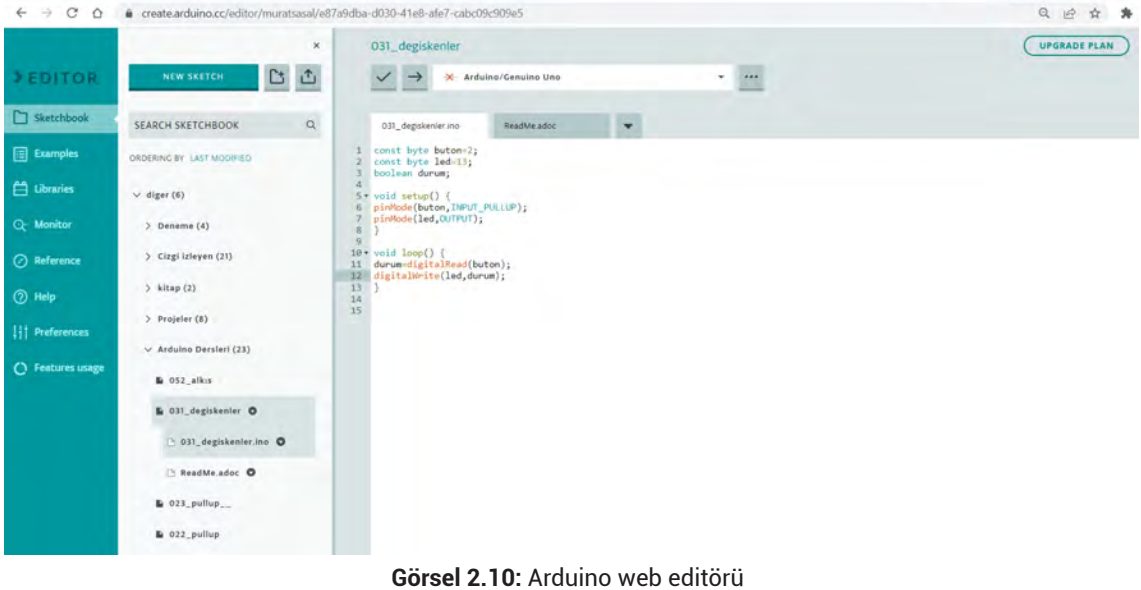
Programlayıcı: Kullanılan programlayıcı tipini gösterir.

Önyükleyici Yazdır: Önyükleyicisi (bootloader) olmayan kartlara bootloader yüklemek için kullanılır.

Yardım Menüsü: www.arduino.cc adresinden ulaşabileceğiniz kaynaklara yönlendirir.

Arduino Web Editör

Arduino IDE yazılımını bilgisayara kurmadan https:// create.arduino.cc/editor sayfasından çevrimiçi olarak da Arduino kartları programlanır. Yazılan programlar web sitesinin bulut saklama alanına kaydolmaktadır (Görsel 2.10).

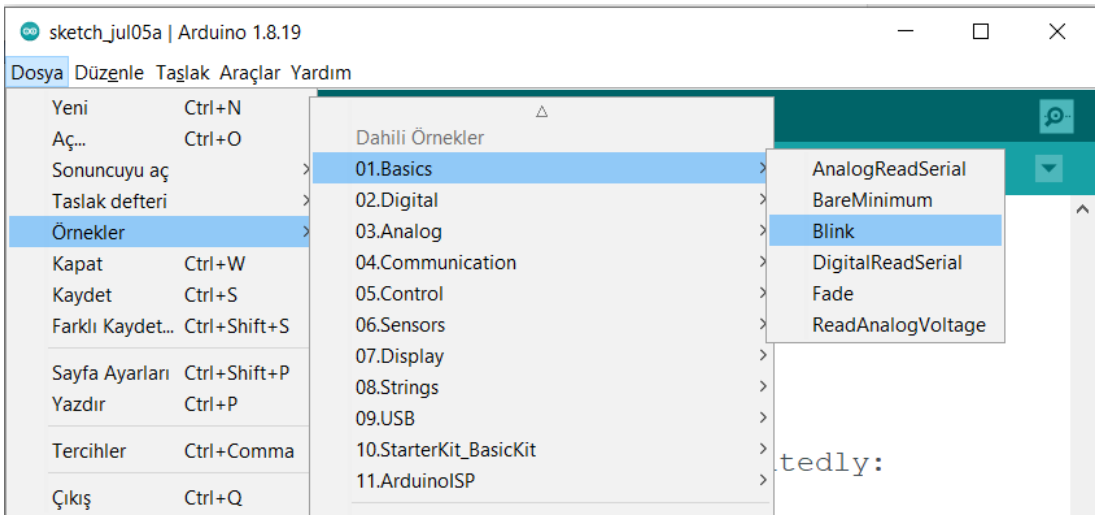


Görsel 2.10: Arduino web editörü

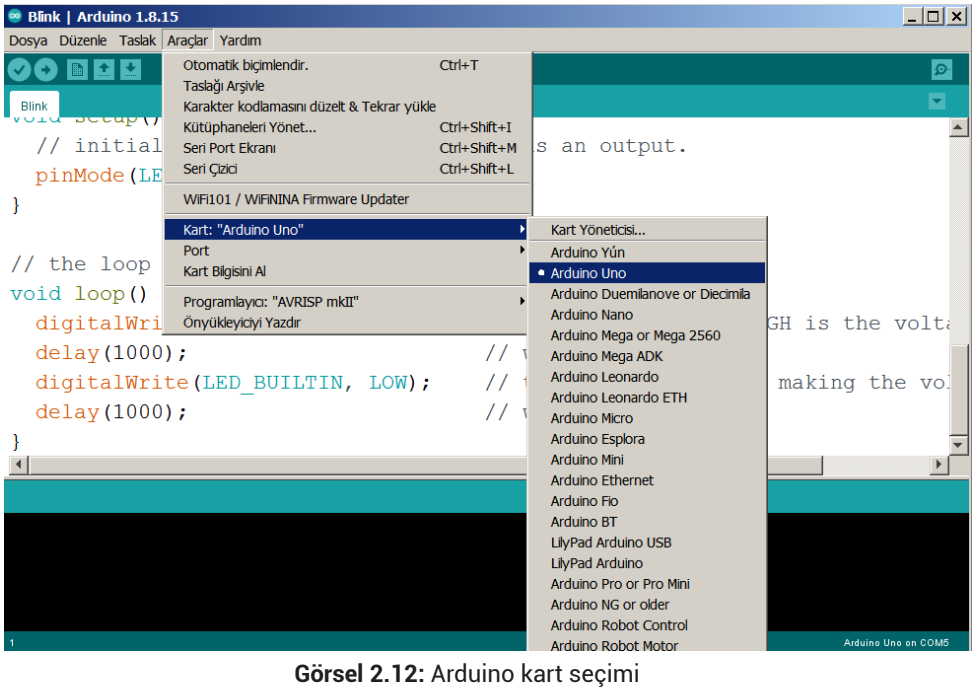
2.4. MİKRODENETLEYİCİYE PROGRAM YÜKLEME

Arduino "Dosya-Örnekler" menüsünde pek çok hazır uygulama mevcuttur. Temel uygulamalardan Blink (yanıp sönen LED) uygulaması Arduino üzerindeki dâhilî LED'i (Bu LED 13 numaralı pine bağlıdır.) 1 saniye aralıkla yakıp söndürmektedir.

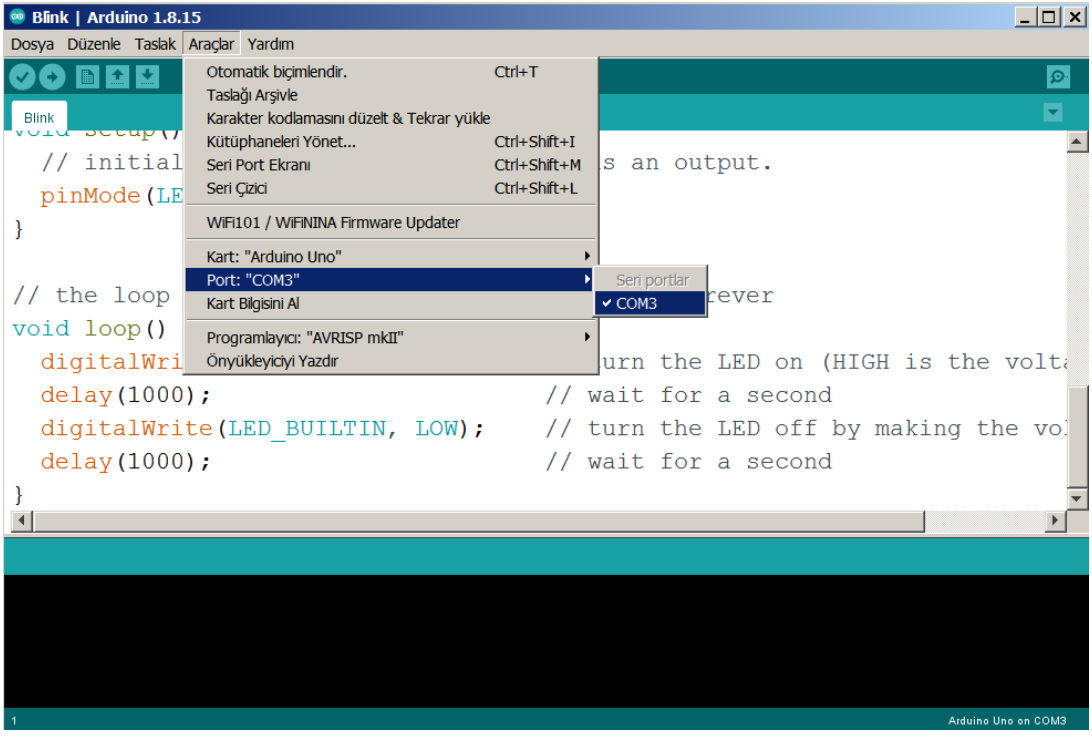
Görsel 2.11'de Dosya → Örnekler → 01.Basics → Blink yolunu izleyerek Blink uygulaması seçilir. Görsel 2.12'de "Araçlar" menüsündeki kart seçeneğinden Arduino Uno (kullanılacak kart çeşidi) seçilir. Görsel 2.13'te "Araçlar" menüsündeki port seçeneğinden Arduino kartımızın kartının bilgisayara bağlı olduğu USB portu seçilir. Görsel 2.14'te "Yükle" düğmesine basılarak Blink uygulaması Arduino kartına yüklenir. Yükleme tamamlandıktan sonra 13 numaralı pinin hizasındaki dâhilî LED birer saniye aralıklarla yanıp söner.



Görsel 2.11: Örnek programlara ulaşma



Görsel 2.12: Arduino kart seçimi



Görsel 2.13: Arduino kartının bağlı olduğu port seçimi



Görsel 2.14: Blink uygulaması

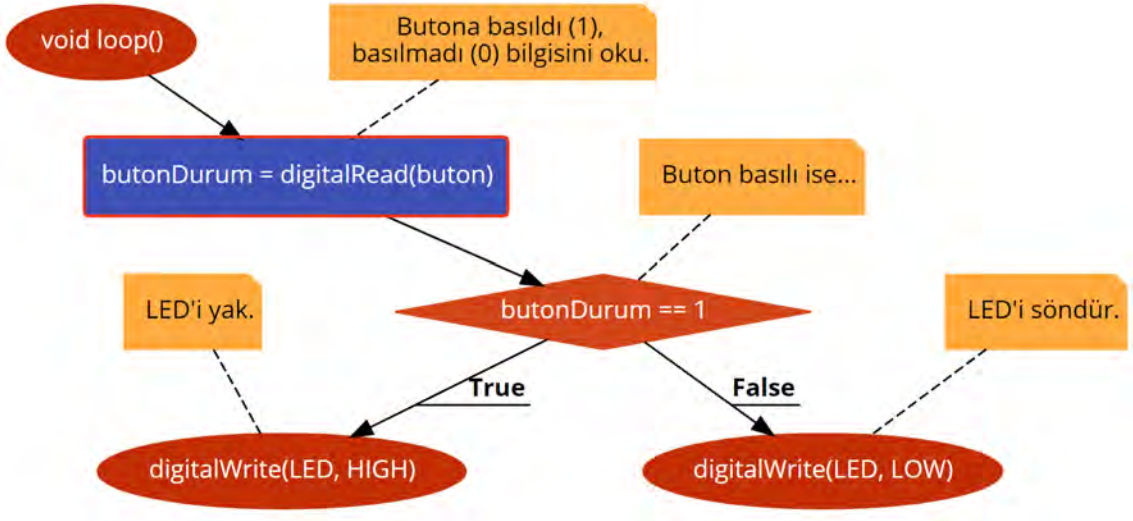
2.5. ALGORİTMA HAZIRLAMA

Algoritma, bir problemi çözmek için takip edilecek sonlu sayıda adımdan oluşan bir çözüm yoludur. Diğer bir ifadeyle algoritma, bir problemin mantıksal çözümünün adım adım nasıl gerçekleştirileceğinin sözlü ifadesidir.

Algoritmayla oluşturulan çözümler sözel olarak ifade edildiğinden algoritmayı herkesin aynı sonucu çıkarabileceği hâle getirmek için akış diyagramları kullanılır. Akış diyagramları sembollerden oluşmaktadır. Her sembolün belli bir işlevi vardır.

Algoritması oluşturulmuş bir problemin bilgisayar ortamına aktarılmış hâline **program** denir. Program, problemin çözümünde yapılması gereken işlemler bütününün kod karşılığıdır. Algoritmaların program hâline getirilmesi için programlama dilleri kullanılır. Programlama dilleri kullanılarak yazılımlar geliştirilir. Arduino programlama dili C/C++ programlama dilinden üretilmiş bir dildir.

Görsel 2.15'te butona basıldığında LED'in yandığı bir olayın algoritması, akış diyagramı ve programı görülmektedir.



Görsel 2.15: Karar yapısı algoritması

2.6. TEMEL PROGRAMLAMA İŞLEMLERİ

Arduino programlama dili, C programlama diline benzemektedir. Her programlama dilinde yapılan temel işlemler Arduino programlama dilinde de yapılmaktadır.

Arduino IDE'de yeni bir taslak oluşturulduğunda taslak (sketch) Görsel 2.16'da görüldüğü gibi iki hazır fonksiyonla gelmektedir. Bunlar setup() (ayar) ve loop() (döngü) fonksiyonlarıdır. Bu fonksiyonlar taslakta bulunmazsa derleyici hata verir.

setup() Fonksiyonu

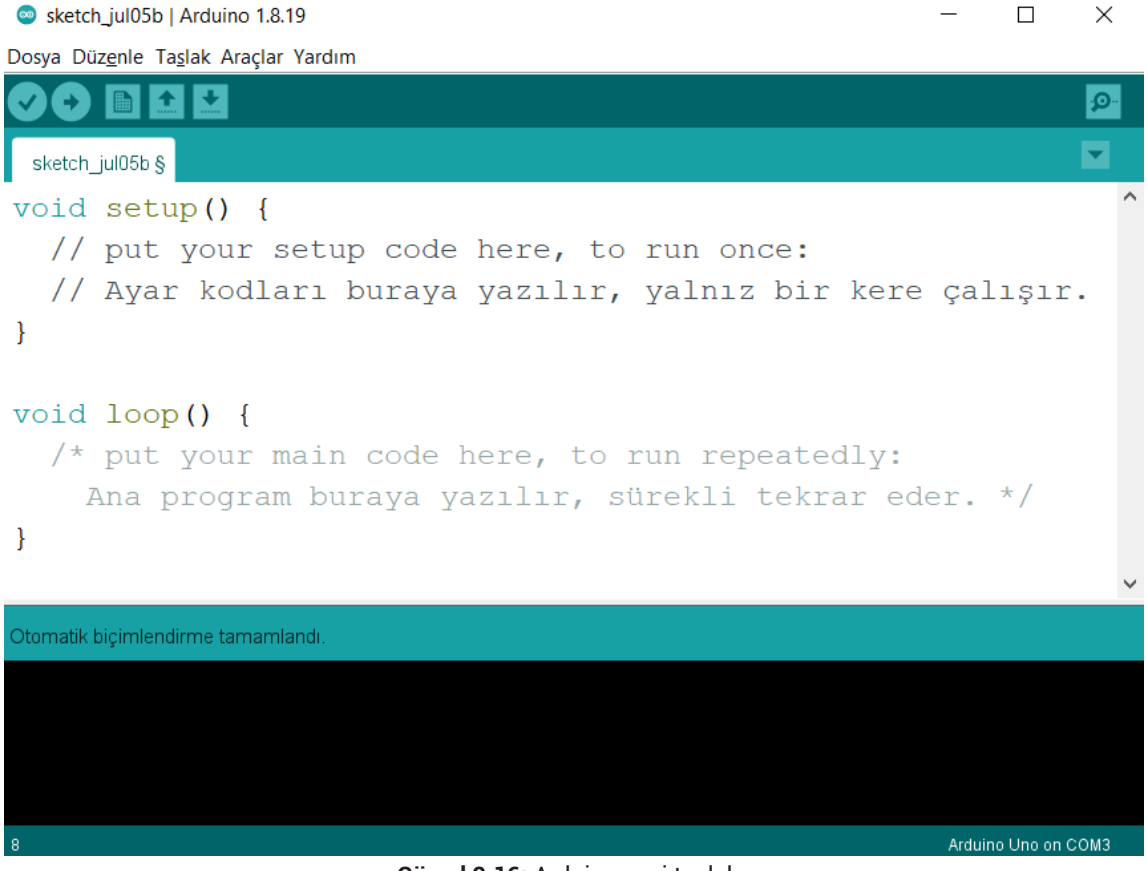
setup() fonksiyonu donanımsal ayarların yapıldığı kart başlatıldığında veya resetlendiğinde bir kez çalışan kısımdır. Void ifadesi setup() fonksiyonun geri herhangi bir değer döndürmediğini ifade etmektedir.

loop() Fonksiyonu

loop() fonksiyonu ana programın olduğu ve kartın enerjisi kesilene kadar sürekli çalışan döngüdür. loop() fonksiyonu da void ifadesiyle yazılır ve geriye değer döndürmez.

" // " ifadesinden sonra aynı satırda yer alan metin açıklama kısımlarıdır ve programın işleyişini etkilemez.

" /* " ve " */ " ifadeleri birden fazla satır içeren açıklama metni yazmak için kullanılır ve programın işleyişini etkilemez.



Görsel 2.16: Arduino yeni taslak

2.7. DİJİTAL GİRİŞ ÇIKIŞ İŞLEMLERİ

Arduino'da dijital işlemler 0 ve 1'in yani 0 V ile 5 V'luk iki seçeneğin olduğu işlemlerdir. Giriş işlemleri Arduino pinlerinin haricî bir devreden elektrik enerjisi alınmasıdır. Çıkış işlemleri ise Arduino pinlerinden haricî bir devreye elektrik enerjisi gönderilmesidir. Arduino'da dijital işlemlerde kullanılan üç temel fonksiyon vardır.

2.7.1. Port Tanımlama pinMode() Fonksiyonu

Arduino'da dijital işlemler 0 ve 1'in yani 0 V ile 5 V'luk iki seçeneğin olduğu işlemlerdir. Giriş işlemleri Arduino pinlerinin haricî bir devreden elektrik enerjisi alınmasıdır. Çıkış işlemleri ise Arduino pinlerinden haricî bir devreye elektrik enerjisi gönderilmesidir. Arduino'da dijital işlemlerde kullanılan üç temel fonksiyon vardır.

Ardino 0 - 13 numaralı dijital pinleriyle A0-A5 (14-19 numaralı) analog pinlerini (Dijital olarak kullanılabilir.) giriş (INPUT) veya çıkış (OUTPUT) olarak ayarlamak için kullanılan fonksiyondur.

pinMode(pin, mod)

pin: Bağlantı ucu numarası.

mod: INPUT, OUTPUT veya INPUT_PULLUP

Örnek: Dijital pinleri giriş veya çıkış olarak ayarlar.

```
pinMode(2, INPUT); // 2 numaralı pin giriş olarak ayarlandı.
pinMode(19, INPUT_PULLUP); // 19 numaralı pin giriş ve pull-up olarak ayarlandı.
pinMode(13, OUTPUT); // 13 numaralı pin çıkış olarak ayarlandı.
```

INPUT olarak yapılandırılmış Arduino (ATmega) pinleri yüksek empedansa ayarlanır. Pinlerin önünde 100 MΩ'luk bir seri direnç bağlanmış şeklinde düşünülebilir. Giriş olarak ayarlanan pinler sensör vb. okumak için kullanışlı hâle gelir. Girişte buton kullanılırsa 10 kΩ'luk pull-up veya pull-down direnci de kullanılmalıdır.

INPUT_PULLUP kullanıldığında Arduino içindeki dâhilî pull-up dirençleri (20 kΩ) aktif edilir. Devrede pull-up direnci kullanmaya gerek yoktur. Giriş sürekli 1'e çekili olduğundan aktif 0 çıkışlı sensörler kullanılmalı veya devre bu şekilde tasarlanmalıdır. INPUT veya INPUT_PULL-UP komutuyla giriş olarak yapılandırılan pinler, 0 V altındaki (negatif voltajlar) veya 5 V üzerindeki voltajlara bağlanırsa zarar görebilir veya bozulabilir. 13 numaralı pinde dâhilî LED ve ona bağlı seri direnç bulunduğundan INPUT_PULLUP kullanıldığında 5 V yerine 1,7 V'ta kalacağından kullanışlı değildir.

OUTPUT olarak yapılandırılan pinler, düşük empedansa ayarlanır. Arduino pinleri, diğer cihazlara 40 mA'e kadar akım sağlayabilir. Bu akım LED gibi düşük güçlü elemanlar için yeterli olsa da motor gibi yüksek güç isteyen elemanlar için araya yeterli akım ve gerilimi sağlayabilecek yükselteç (sürücü) devreleri ilave edilmelidir. OUTPUT olarak ayarlanan pinler, GND'ye veya 5 V'a bağlanırsa hasar görebilir veya bozulabilir.

NOT pinMode() fonksiyonu loop() fonksiyonunun içinde kullanılarak bir pin, programın belli kısımlarında INPUT, belli kısımlarında OUTPUT olarak ayarlanabilir. Kullanılan dijital pin sayısını azaltmak için bu yönteme başvurulabilir.

2.7.2. Dijital Çıkış İşlemleri

digitalWrite() Fonksiyonu

pinMode() fonksiyonuyla OUTPUT olarak ayarlanan pinlere HIGH (5 V veya lojik 1) veya LOW (0 V veya lojik 0) vermek için kullanılır. Örneğin pine 5 V verildiğinde akım kaynağı olarak katodu seri bir dirençle GND'ye bağlı olan LED ışık verir. 0 V verildiğinde anodu seri bir dirençle 5 V'a bağlanmış LED ışık verir.

digitalWrite(pin, deger)

pin: Pin numarası

deger: HIGH veya LOW (1 veya 0)

Örnek: Dijital çıkışı 5 V veya 0 V yapar.

```
digitalWrite(13, HIGH); // 13 numaralı pin çıkışı 1 olur.
digitalWrite(13, LOW); // 13 numaralı pin çıkışı 0 olur.
```

Arduino UNO'da A0 – A5 olarak adlandırılan altı adet analog giriş pini, dijital pin olarak da kullanılabilir. Böylelikle Arduino UNO'da yirmi adet pin digital işlemler için kullanılabilir. İstisna olarak Arduino Nano, Pro Mini ve Mini'nin A6 ve A7 pinleri yalnızca analog giriş olarak kullanılmaktadır.

NOT Bir pin pinMode() fonksiyonuyla OUTPUT olarak ayarlanmadığında ve o pine bağlı LED'e, digitalWrite() fonksiyonuyla HIGH gönderildiğinde LED sönük görünebilir. Bunun nedeni PinMode() fonksiyonu ayarlamadan digitalWrite() kullanılması, o pinde akım sınırlayıcı direnç gibi davranan dahilî pull-up direncini etkinleştirmesidir.

delay() Fonksiyonu

Programı, parametre olarak belirtilen süre kadar (milisaniye cinsinden) duraklatır (1000 milisaniye 1 saniyedir.). Delay fonksiyonu işlemleri durduğu için diğer işlemler sensör okuma vb. yapılmayacaktır. delay() fonksiyonu, işletilen programı delay fonksiyonun olduğu satırda belirtilen süre boyunca bekletir. Bu sırada bir sonraki komuta geçilemez. Diğer hiçbir işlem yapılmaz. Delay fonksiyonu işlemleri durdurduğu için diğer işlemler sensör okuma vb. yapılmayacaktır. Gelişmiş bir programda delay() yerine milis() fonksiyonu kullanılır.

delay(ms); //Programı bekletme komutu yazılımı.

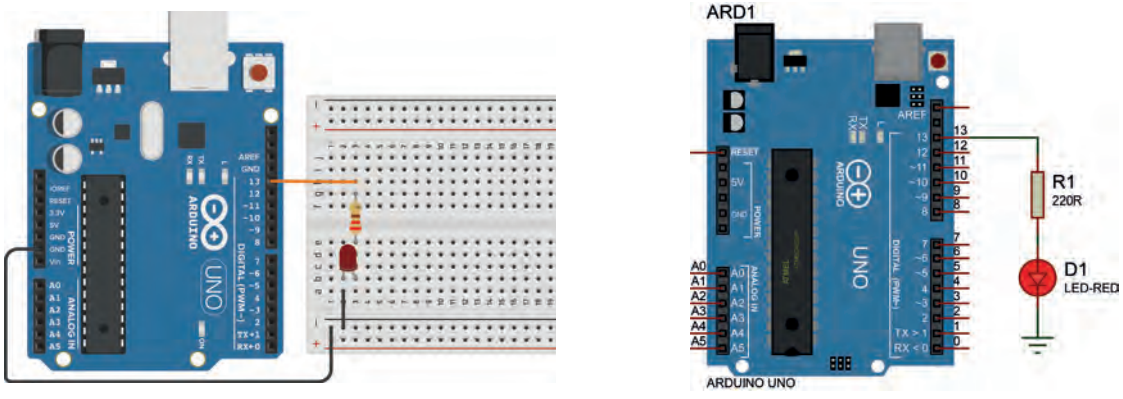
ms: Duraklatılacak sürenin milisaniye cinsinden sayısı. Veri tipi: unsigned long.

delay(500);//Programı 500 ms (yarım dakika) bekletir.

Uygulama Adı: LED'le Dijital Çıkış Uygulaması — No.: 1

Amaç: LED'le dijital çıkış uygulaması yapmak.

Görsel 2.17'de Arduino'nun 13. pininde bir LED ve LED'e seri bağlı direnç vardır. Arduino kodu LED'i 1 saniye aralıklarla yakıp söndürmektedir.



Görsel 2.17: LED'le dijital çıkış şeması ve devresi

LED'le dijital çıkış uygulaması programı aşağıdaki gibidir:

```
void setup() {          // Portların giriş-çıkış olarak tanımladığı fonksiyon. (Bir
kez çalışır.)
pinMode(13, OUTPUT);    // 13 numaralı pin çıkış olarak ayarlandı.
}
void loop() {           // Loop sonsuz döngüsü. (Sürekli çalışır.)
digitalWrite(13, HIGH); // 13 numaralı pinden 5 V (lojik 1) gönderildi.
delay(1000);            // Zaman gecikmesi // Bekle 1000 ms = 1 sn.
digitalWrite(13, LOW);  // 13 numaralı pin 0 V'a (GND) çekildi. (lojik 0)
delay(1000);            // Zaman gecikmesi // Bekle 1000 ms = 1 sn.
}
```

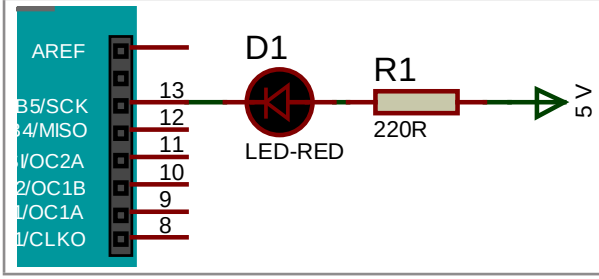
İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.17'deki devreyi kurunuz.
4. Programı yazıp Arduino'ya yükleyiniz. LED'in çalışmasını gözlemleyiniz.
5. digitalWrite(13, HIGH); komut satırını digitalWrite(13, 1); olarak, digitalWrite(13, LOW); komut satırını digitalWrite(13, 0); olarak değiştirip programı yükleyiniz.
6. Programdaki delay(1000); fonksiyonlarının parametresini değiştirip LED'in durumunu gözlemleyiniz.
7. A0 pinini kullanarak programı ve devreyi yeniden düzenleyiniz. Programı yükleyerek devrenin çalışmasını gözlemleyiniz.

8. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

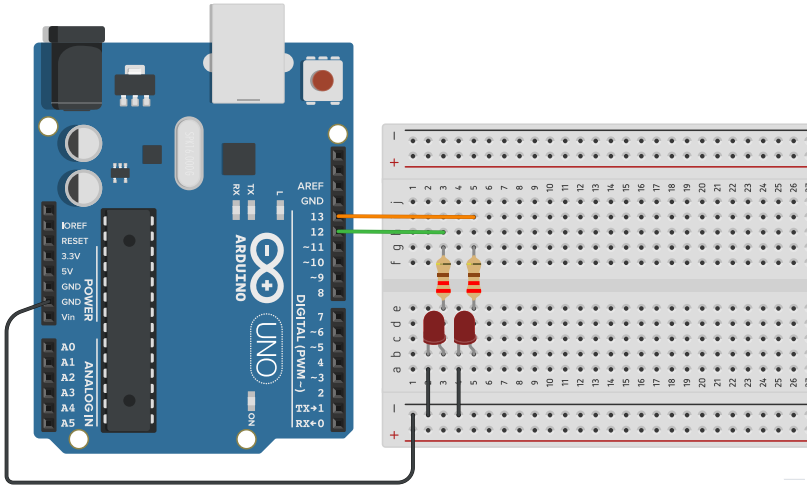
SIRA SİZDE

1. Devre bağlantısını altta verilen şekildeki gibi düzenleyip digitalWrite(13, 0); komutuyla lojik 0 göndererek LED'in yandığını gözlemleyiniz.




```
void setup() {
 pinMode(13, OUTPUT);
}
 void loop() {
 digitalWrite(13, 0);// LED'i 2 saniye yak.
 delay(2000);
 digitalWrite(13, 1);// LED'i 0,5 saniye söndür.
 delay(500);
}
```

2. Arduino'nun 12 numaralı pinine ikinci bir LED bağlayarak programı iki LED sırayla yanıp sönecek şekilde yeniden düzenleyiniz.



Sorular

1. delay(100); fonksiyonunun parametresi "100"'den küçük değerler girildiğinde LED nasıl görünmektedir? Nedenini açıklayınız.

2. LED'e seri bağlı direnç niçin kullanılmıştır? Yazınız.

3. Arduino Uno'da kaç adet pin dijital giriş çıkış olarak ayarlanabilir? Belirtiniz.

Değişkenler

Değişkenler, içerisinde veri saklamak için kullanılan bellekte en az bir byte yer kaplayan ifadelerdir. int a = 5; komutuyla "a" isminde integer (tam sayı) bir değişken oluşturulmuş ve içerisine 5 verisi (sabiti) yüklenmiştir. "a" değişkeninin içeriği program içerisindeki çeşitli işlemlerle (matematiksel, port giriş bilgileri vb.) değişebilir. Değişkenler farklı veri tiplerinde tanımlanır. Veri tipine göre hafızada kapladıkları alan değişir.

Değişkenlere isim verme kuralları şunlardır:

1. Bir değişkenin ilk harfi rakam olamaz.

2. Değişken isimlerinde alt tire "_" hariç özel karakter bulunamaz.

3. Türkçe karakter kullanılamaz.

4. Boşluk içermez.

5. Arduino komutları (int, loop, float vb.) değişken adı olamaz.

6. Küçük büyük harf duyarlıdır. Örneğin "LED" ile "led" farklı değişken isimleridir.

Programın okunabilirliği için değişken isimleri anlamlı olmalıdır. Örneğin butonun durumunu kontrol etmek için butonDurum adında bir değişken oluşturulabilir. Değişken adı birden fazla kelimeden oluşuyorsa ya kelimeler arası alt çizgi kullanılır ya da daha çok tercih edilen ve kabul gören şekliyle baştaki kelime hariç diğer kelimelere büyük harfle başlanır. "ledPin", "triggerPinDurum", "potDegeri" gibi değişken isimleri Arduino söz dizimine benzer okunabilirliği yüksek örnek değişken isimleridir. Tablo 2.1'de değişken tipleri ve özellikleri görülmektedir.

Tablo 2.1: Değişken Tipleri ve Özellikleri

Değişken tipi	Hafızada kapladığı alan	Örnek	Açıklama
byte	1 byte	byte LED = 13; byte a = 0, b = 128, c=255;	0-255 arası işaretsiz sayılar

boolean (bool)	1 byte	bool butonDurum=0;
		boolean x = 0, y = 1;
		boolean kontrolBiti = True;
			0 (FALSE) veya 1 (TRUE) değerlerini alır.

char	1 byte	char karakter = 'A'; char karakter = 65; char karakterDizisi[] = "Arduino";	-128 ile 127 arasındaki değerleri alır.
unsigned char	1 byte	unsigned char a = 0, b = 255;	0-255 arası işaretsiz sayılar.
string	1 byte	string metin = "Arduino";	Metin işlemleri için kullanılır. (char dizisi)
int	2 byte	int a = 5, b = -618, c = 20140;	İşaretli sayılar için-32.768-32.767
unsigned int	2 byte	unsigned int sayim = 50000;	İşaretsiz sayılar için 0-65,535
word	2 byte	word sure = 60000;	İşaretsiz sayılar için 0-65,535

short	2 byte	short a = -272, b = 32000;	İşaretli sayılar için -32.768-32.767
long	4 byte	long zaman = -86400000;	İşaretli sayılar için -2.147.483.648-2.147.483.647
unsigned long	4 byte	unsigned long sure = 604800000;	İşaretsiz sayılar için 0-4.294.967.295
float	4 byte	float pi = 3.14; float oran = 0.618;	İşaretli ondalıklı sayılar -3,4028235E+38-3,4028235E+38
double	4 byte	double e = 2,718281; double f = 1.618;	İşaretli ondalıklı sayılar -3,4028235E+38-3,4028235E+38
			Due kartında 64 bitlik (8 byte) yer tutar.

NOT

Boolean veri tipinde "false" ifadesi sadece sıfır için geçerliyken " true" ifadesi sıfır dışındaki tüm sayılar için geçerlidir.

Değişkenlerin Dönüşümü

Değişken tipleri arasında işlem yapılırken bir değeri başka bir değere dönüştürmek gerekebilir. Bu dönüşüm işlemlerini derleyici bazen otomatik olarak yapsa da bazı zamanlarda dönüşümün programı yazan tarafından yapılması gerekebilir. Tablo 2.2'de değişkenlerin nasıl dönüştürüleceği görülmektedir.

Tablo 2.2: Değişkenlerin Dönüştürülmesi

char()	Parantez içerisine yazılan herhangi bir veriyi "char" veri türüne dönüştürür.
byte()	Parantez içerisine yazılan herhangi bir veriyi "byte" veri türüne dönüştürür.
int()	Parantez içerisine yazılan herhangi bir veriyi "int" veri türüne dönüştürür.
long()	Parantez içerisine yazılan herhangi bir veriyi "long" veri türüne dönüştürür.
float()	Parantez içerisine yazılan herhangi bir veriyi "float" veri türüne dönüştürür.

Değişken Niteleyiciler

Static: Sadece tanımlandığı fonksiyonda içeriği değişebilir. Diğer fonksiyonlar bu değişkenin içeriğini değiştiremez.

Const: Değişkenin içeriğinin hiçbir şekilde değiştirilemeyeceğini belirtir.

Volatile: Tanımlanan değişkenler Arduino'nun RAM bölgesine kaydedilir. Sadece interruptla değişir.

Değişkenlerin Kapsamı

Bir değişken tanımlandığı yere göre programın tamamında, belli bir fonksiyonda hatta tek bir döngüde faaliyet gösterebilir.

Global Değişkenler: Herhangi bir fonksiyonun içinde değil de en üstte tanımlanma blokunda tanımlanan değişkenlerdir. Program bitene kadar bellekten alınan alan geri verilmez ve değişkenin değeri korunmaya devam eder. Bütün fonksiyonlarda kullanılabilir. Yerel fonksiyonlarda aynı isimle farklı değişken oluşturulamaz.

Local Değişkenler: Bir blok veya fonksiyon içinde tanımlanır. Sadece tanımlandıkları alanda kullanılır. Bir fonksiyonda tanımlandılarsa o fonksiyon bittikten sonra alınan alan belleğe geri verilir. Farklı fonksiyonlar içinde aynı ada sahip yerel değişkenler tanımlamak mümkündür.

Örnek: Değişkenlerin faaliyet alanını gösterir.

```
int sayi;  // Tüm fonksiyonlar sayi değişkenini kullanabilir.
void setup() {
  // ...
}
void loop() {
  int i;    // “i” değişkeni sadece loop fonksiyonu içinde kullanılabilir.
  float f;  // “f” değişkeni sadece loop fonksiyonu içinde kullanılabilir. // ...
  for (int j = 0; j < 100; j++) {
    // “j” değişkeni sadece bu for döngüsü içinde kullanılır.
  }
}
```

Değişken Araçları

PROGMEM: Bilgiyi SRAM bellek yerine flash bellekte depolamaya yarar. Arduino programında pek çok char tanımlanırsa SRAM yetersiz kalabilir. Bu yüzden çok uzun metinler PROGMEM ile flash belleğe kaydedilir.

sizeof(): Değişkenlerin veya dizilerin byte sayısını verir.

Sabitler

Sabitler, Arduino dilinde önceden tanımlanmış ifadelerdir. Programların okunmasını kolaylaştırmak için kullanılır. Sabitler değişken ismi olarak kullanılamaz.

Mantıksal düzeyleri tanımlama: true ve false (Boolen sabitleri)

Pin seviyelerini tanımlama: HIGH ve LOW

Dijital pin modlarını tanımlama: INPUT, INPUT_PULLUP ve OUTPUT

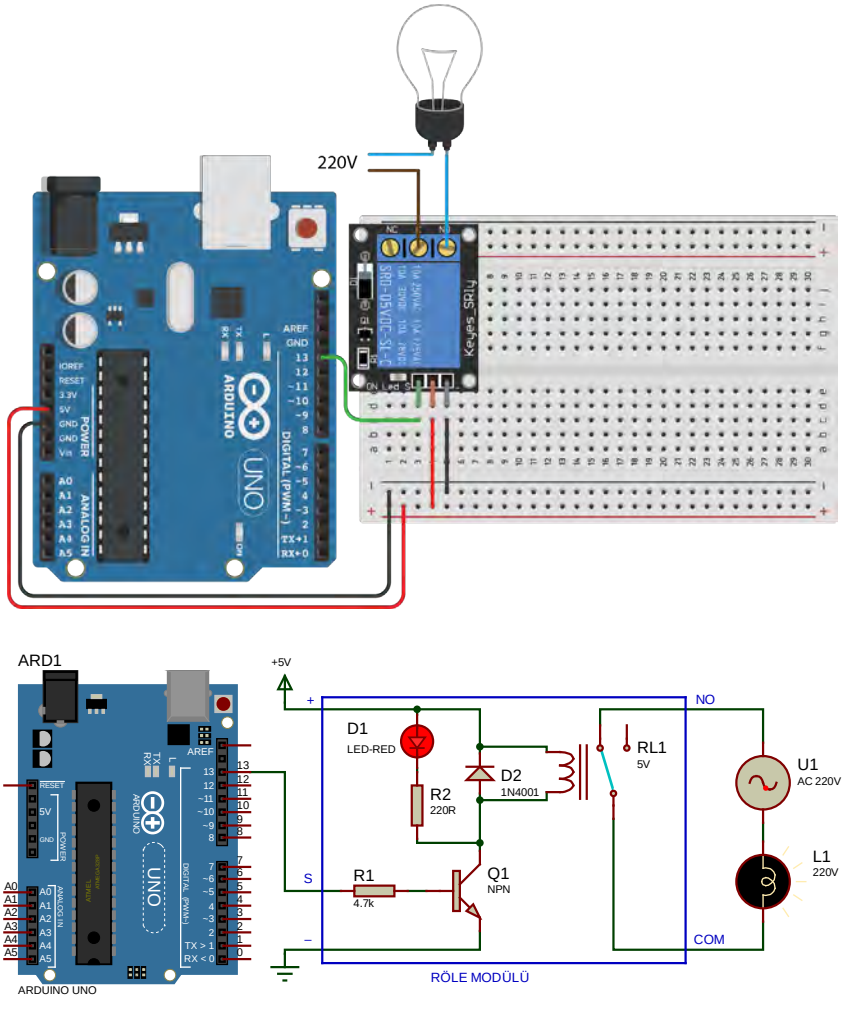
Dâhilî LED (13 numaralı pine bağlı): LED_BUILTIN

Uygulama Adı: Röleyle Dijital Çıkış Uygulaması — No.: 2

Amaç: Röleyle dijital çıkış uygulaması yapmak.

Görsel 2.18'de röle modülünün beslemesi Arduino kartının 5 V ve GND uçlarından verilir. Modülün sinyal girişine Arduino'nun çıkış olarak ayarlanan 13 numaralı pini bağlanır. Modülün NO-COM uçlarına 220 V'la seri bağlanan lamba normalde sönüktür. Modülün sinyal girişi Arduino'nun 13 numaralı pininden lojik 1 (5 V) bilgisini aldığında röle içinde COM ucu NO kontağıyla temas eder ve lamba yanar. Modülün sinyal girişi Arduino'nun 13 numaralı pininden lojik 0 (GND) bilgisini aldığında röle içinde COM ucu NO kontağından ayrılır (NC kontağıyla temas eder.) ve lamba söner.

Arduino'nun 13 numaralı pini **byte** tipinde **role** değişkenine atanmıştır (**role** ifadesinin içine 13 sayısı yüklenmiştir.). Program içerisinde 13 sayısı yerine **role** ifadesi (değişkeni) kullanılır. **role** değişkeninin içeriği program içinde değişmesi istenmediğinden **const** ifadesi kullanılmıştır.



Görsel 2.18: Röleyle dijital çıkış uygulama şeması ve devresi

Röleyle dijital çıkış uygulama programı aşağıdaki gibidir:

```
const byte role = 13;      // Arduino'nun 13 nolu ucu röle değişkenine atandı.
void setup() {             // Setup ayarları // Portların giriş-çıkış olarak tanımladığı bölüm.
pinMode(role, OUTPUT);     // Role, yani 13 nolu pin çıkış olarak tanımlandı.
}
void loop() {              // Loop sonsuz döngüsü.
digitalWrite(role, HIGH); // Role değişkenini HIGH (1) yap.
delay(1000);               // Zaman gecikmesi // Bekle 1000 ms = 1 sn
digitalWrite(role, LOW);  // Role değişkenini LOW (0) yap.
delay(1000);               // Zaman gecikmesi // Bekle 1000 ms = 1 sn
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.18'deki devreyi kurunuz. Devre bağlantısında 220 V bulunduğundan devreye enerji vermeyiniz.
4. Programı yazıp Arduino'ya yükleyiniz.
5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

DİKKAT

Devre bağlantısında 220 V bulunduğundan devreye enerji vermeyiniz. Devreyi çalıştırmak için öğretmeninizden yardım isteyiniz.

3. Öğretmeniniz gözetiminde devrenin çalışmasını gözlemleyiniz.

SIRA SİZDE

1. delay(sure); şeklindeki bir komutu kullanabilmek için programın başındaki değişken tanımlamasını yapınız.
2. LED ile dijital çıkış uygulamasının programını değişken kullanarak yazınız.

Soru

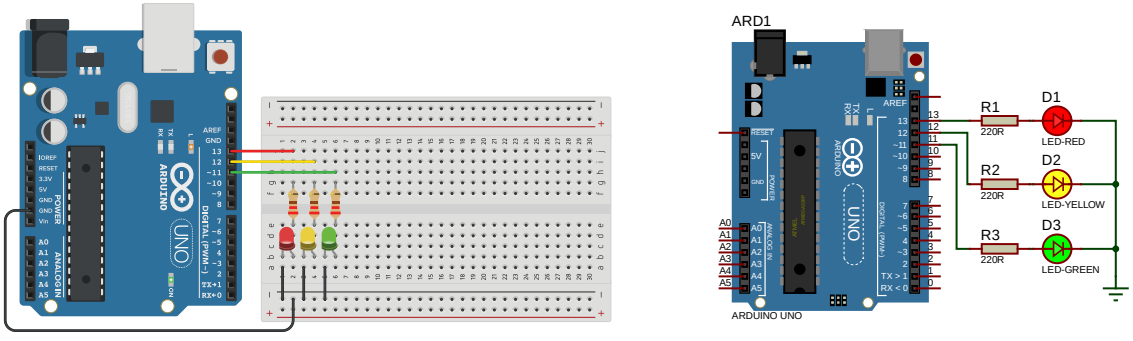
1. digitalWrite(13,1); komutu programdaki hangi satırla eş değerdir? Belirtiniz.

Uygulama Adı: Trafik Lambası Uygulaması — No.: 3

Amaç: Trafik lambası uygulaması yapmak.

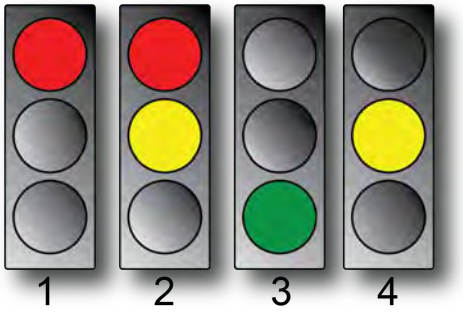
Görsel 2.18'de röle modülünün beslemesi Arduino kartının 5 V ve GND uçlarından verilir. Modülün Görsel 2.19'da 13, 12, 11 numaralı pinlere sırasıyla kırmızı, sarı ve yeşil LED bağlanarak programda çıkış olarak ayarlanmıştır. Her durum için yanan lambalara lojik 1 bilgisi gönderilirken sönmesi istenen lambalara lojik 0 bilgisi gönderilir.

Değişkenler (kirmizi, sari, yesil) **byte** (0-255 arası sayılar için) tipinde tanımlanarak **const** ifadesiyle içeriği değiştirilmeyecek şekilde ayarlanmıştır.



Görsel 2.19: Trafik lambası uygulama şeması ve devresi

Görsel 2.20'de trafik lambası çalışma sırası görülmektedir. Kırmızı ve yeşil ışık için 3 saniye, sarı ışık için 1 saniye yanma süresi verilmiştir.



Görsel 2.20: Trafik lambası çalışma sırası

Trafik lambası uygulama programı aşağıdaki gibidir:

```
const byte kirmizi = 13, sari = 12, yesil = 11;
void setup() {
  pinMode(kirmizi, OUTPUT); // Kırmızı LED.
  pinMode(sari, OUTPUT); // Sarı LED.
 pinMode(yesil, OUTPUT); // Yeşil LED.}
void loop() {
```

```
 digitalWrite(kirmizi, 1); // Kırmızı LED'i yak.
  digitalWrite(sari, 0);
  digitalWrite(yesil, 0);
  delay(3000);

  digitalWrite(kirmizi, 1); // Kırmızı LED'i yak.
  digitalWrite(sari, 1); // Sarı LED'i yak.
  digitalWrite(yesil, 0);
  delay(1000);

  digitalWrite(kirmizi, 0);
  digitalWrite(sari, 0);
  digitalWrite(yesil, 1); // Yeşil LED'i yak.
  delay(3000);

  digitalWrite(kirmizi, 0);
  digitalWrite(sari, 1); // Sarı LED'i yak.
  digitalWrite(yesil, 0);
  delay(1000);
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.19'daki devreyi kurunuz.
4. Programı yazıp Arduino'ya yükleyerek devrenin çalışmasını gözlemleyiniz.
5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. boolean yak=1, sondur=0; şeklinde değişkenler tanımlayarak LED'e 1 ve 0 bilgisini "yak" ve "sondur" değişkenleriyle gönderiniz.

```
// …
void loop() {
  digitalWrite(kirmizi, yak); // Kırmızı LED'i yak.
  digitalWrite(sari, sondur);
  digitalWrite(yesil, sondur);
  delay(3000);
// …
```

Operatörler

İşlem yapmayı sağlayan işaretlere **operatör** denir. Tablo 2.3'te aritmetiksel, Tablo 2.4'te bileşik operatörler, Tablo 2.5'te karşılaştırma operatörleri, Tablo 2.6'da mantıksal (lojik) operatörler görülmektedir.

Tablo 2.3: Aritmetiksel Operatörler

İşlem	Operatör	Örnek	Açıklama
Toplama	+	a+b	a ile b'yi topla.
Çıkarma	-	a-b	a'dan b'yi çıkar.
Çarpma	*	a*b	a ile b'yi çarp.
Bölme	/	a/b	a'yı b'ye böl.
Mod	%	a%b	a'nın b'ye bölümünden kalan
Atama	=	a = b	b'yi a'ya ata (b'nin içeriğini a'ya yükle.).

Tablo 2.4: Bileşik Operatörler

İşlem	Operatör	Örnek	Açıklama
Bir arttır.	++	a++	a = a + 1
Bir azalt.	--	a--	a = a - 1
n arttır.	+=	a+= n	a = a + n
n azalt.	-=	a-= n	a = a-n
n'yle çarp.	*=	a*= n	a = a * n
n'ye böl.	/=	a/= n	a = a / n
n'ye göre mod	%=	a%= n	a = a % n

NOT

" ++" ve "--" operatörlerinin iki farklı kullanım şekli vardır.
y = x++; // x'i, y'ye yükler, sonra x'i bir artırır.
y = ++x; // x'i bir artırır, sonra x'i, y'ye yükler.

Örnek: ++'ın değişkenin önünde ve arkasında kullanımını gösterir.

```
x = 2;
y = ++x;  // x'in içeriği 3, y'nin içeriği 3.
y = x++;  // x'in içeriği 4 ancak y'nin içeriği hâlâ 3.
```

Tablo 2.5: Karşılaştırma Operatörleri

İşlem	Operatör	Örnek	Açıklama
Eşit mi?	==	a == b	a b'ye eşit mi?
Eşit değil mi?	!=	a != b	a b'ye eşit değil mi?
Büyük mü?	>	a > b	a b'den büyük mü?
Küçük mü?	<	a < b	a b'den küçük mü?
Büyük eşit mi?	>=	a >= b	a b'den büyük eşit mi?
Küçük eşit mi?	<=	a <= b	a b'den küçük eşit mi?

Tablo 2.6: Mantıksal (Lojik) Operatörler

İşlem	Operatör	Örnek	Açıklama
Ve (And)	&&	a>b && a>c	a, b'den ve c'den büyük mü?
Veya (Or)	||	a>b || a<c	a, b'den büyük veya c'den küçük mü?
Değil (Not)	!	!(a>b)	a, b'den büyük değil mi?

Bitsel Operatörler

Bitsel operatörler değişken ve veriler üzerinde bit seviyesinde ve, veya, değil, xor, bit kaydırma gibi işlemler yapmak için kullanılan operatörlerdir. Tablo 2.7'de a=60 ve b=13 ondalık sayıları için bit düzeyinde işlem örnekleri verilmiştir.

a = 111100 → 60 sayısının binary karşılığı

b **= 001101** → 13 sayısının binary karşılığı

&_____

Q = 001100 → a & b = 12 sayısının binary karşılığı

Tablo 2.7: a=60 ve b=13 Ondalık Sayıları İçin Bit Düzeyinde İşlem Örnekleri

İşlem	Operatör	Örnek (a=60 b=13 için)	Açıklama
Ve (And)	&	(a & b) onluk 12 ikilik 0000 1100	Her iki bit 1 ise sonuç 1'dir.
Veya (Or)	|	(a | b) onluk 61 ikilik 0011 1101	Bitlerden biri 1 ise sonuç 1'dir.
Değil (Not)	~	(~a) onluk -60 ikilik 1100 0011	Sonuç her bitin tersidir.
Özel Veya (Xor)	^	(a ^ b) onluk 49 ikilik 0011 0001	Bitler farklıysa sonuç 1'dir.
Sola Kaydır (Shift Left)	<<	a << 2 onluk 240 ikilik 1111 0000	Operatörün sağındaki sayı kadar biti sola kaydırır (Sağa sayı kadar 0 eklenir.).
Sağa Kaydır (Shift Right)	>>	a >> 2 onluk 15 ikilik 0000 1111	Operatörün sağındaki sayı kadar biti sağa kaydırır (Sola sayı kadar 0 eklenir.).

Örnek: İki girişli lojik ifadeleri çıkışa bağlı LED'lerle gösterir.

```
boolean A; // A ve B iki girişli kapılar.
boolean B;
boolean AND;
boolean OR;
boolean A; // A ve B iki girişli kapılar.
boolean B;
boolean AND;
boolean OR;
boolean NAND;
```

```
boolean NOR;
boolean XOR;
boolean XNOR;
int butonA = 2; // Giriş butonları.
int butonB = 3;
int ledA = 4;// Giriş butonlarının durumunu gösteren LED'ler.
int ledB = 5;
int ledAND = 6;// Çıkış LED›leri.
int ledOR = 7;
int ledNAND = 8;
int ledNOR = 9;
int ledXOR = 10;
int ledXNOR = 11;
void setup() {
  pinMode(butonA, INPUT);
  pinMode(butonB, INPUT);
  pinMode(ledA, OUTPUT);
  pinMode(ledB, OUTPUT);
  pinMode(ledAND, OUTPUT);
  pinMode(ledOR, OUTPUT);
  pinMode(ledNAND, OUTPUT);
  pinMode(ledNOR, OUTPUT);
  pinMode(ledXOR, OUTPUT);
  pinMode(ledXNOR, OUTPUT);
}
void loop() {
  A = digitalRead(butonA);
  B = digitalRead(butonB);
  digitalWrite(ledA, A);
  digitalWrite(ledB, B);
 XNOR = !(A ^ B);
  XOR = A ^ B;
  NOR = !(A | B);
  OR = A | B;
  NAND = !(A & B);
  AND = A & B;
  digitalWrite(ledXNOR, XNOR);
  digitalWrite(ledXOR, XOR);
  digitalWrite(ledNOR, NOR);
  digitalWrite(ledOR, OR);
  digitalWrite(ledNAND, NAND);
  digitalWrite(ledAND, AND);
  delay(50);
```

```
boolean NOR;
boolean XOR;
boolean XNOR;
int butonA = 2; // Giriş butonları.
int butonB = 3;
int ledA = 4;// Giriş butonlarının durumunu gösteren LED'ler.
int ledB = 5;
int ledAND = 6;// Çıkış LED›leri.
int ledOR = 7;
int ledNAND = 8;
int ledNOR = 9;
int ledXOR = 10;
int ledXNOR = 11;
void setup() {
  pinMode(butonA, INPUT);
  pinMode(butonB, INPUT);
  pinMode(ledA, OUTPUT);
  pinMode(ledB, OUTPUT);
  pinMode(ledAND, OUTPUT);
  pinMode(ledOR, OUTPUT);
  pinMode(ledNAND, OUTPUT);
  pinMode(ledNOR, OUTPUT);
  pinMode(ledXOR, OUTPUT);
  pinMode(ledXNOR, OUTPUT);
}
void loop() {
  A = digitalRead(butonA);
  B = digitalRead(butonB);
  digitalWrite(ledA, A);
  digitalWrite(ledB, B);
 XNOR = !(A ^ B);
  XOR = A ^ B;
  NOR = !(A | B);
  OR = A | B;
  NAND = !(A & B);
  AND = A & B;
  digitalWrite(ledXNOR, XNOR);
  digitalWrite(ledXOR, XOR);
  digitalWrite(ledNOR, NOR);
  digitalWrite(ledOR, OR);
  digitalWrite(ledNAND, NAND);
  digitalWrite(ledAND, AND);
  delay(50);
}
```

Döngüler

Koşul sağlanana kadar tekrar edilmesi istenen komutlar için kullanılır. Döngülerde koşul her zaman boolean (true-false) bir ifadedir.

While Döngüsü

While döngüsünde program koşula bakar, koşul doğruysa döngü içine girer ve koşul yanlış olana kadar içerdeki işlemleri tekrarlar. Eğer içeride koşulu bozacak (false) değişiklikler yapılmazsa koşul sürekli doğru olacağından program sonsuz döngüye girer. Döngü içindeki kodlarda koşulu bozacak durum, bir artırım veya sensör bilgisi gibi haricî bir durum olur.

```
while(koşul) {
// Tekrarlanan işlemler.
}
```

Örnek: Program dâhilî LED'i beş kez yakıp söndürür.

```
void setup() {
  int sayac = 0; // Döngüden önce sayaç değişkeninin içeriği 0.
  while (sayac < 5) {
    digitalWrite(13, HIGH); // Yak.
    delay(200);
    digitalWrite(13, LOW); // Söndür.
    sayac++; // Sayaç değişkenini 1 artır.
  }
}
void loop() {
}
```

Örnek: Program döngü içinde sensörün durumunu okumaktadır.

```
while (sensorDurum) { // Sensörden gelen bilgi 1 olduğu sürece döngüde kal.
  digitalWrite(role, 1); // Röleye enerji ver.
  sensorDurum = digitalRead(sensor); /Sensörü oku.
}
```

do...while Döngüsü

While döngüsüyle benzerdir. Farkı do...while döngüsünde önce döngü içindeki işlem yapılır. Sonra koşula bakılır. Komutlar bir kez mutlaka yapılmış olur.

```
do {
// Tekrarlanan işlemler.
}
while(koşul);
```

Örnek: Program dâhilî LED'i on kez yakıp söndürür.

```
i = 1;
do {
digitalWrite(13, HIGH);
```

```
delay(1000);
digitalWrite(13, LOW);
delay(1000);
i++;
}while(i <= 10);
```

Örnek: Program döngü içinde sensörün durumunu okumaktadır.

```
int x = 0;
do {
   x = sensorOku();  // Sensörü kontrol et.
} while (x < 100);
```

For Döngüsü

For ifadesi koşul sağlanana kadar kodların tekrarlanması için kullanılır. Başlangıç değeri, koşul ve artırım değeri olarak üç bölümden oluşur.

```
for(Başlangıç; Koşul; Artırım) {
// Tekrarlanan işlemler.
}
```

Başlangıç: Başlangıçta bir kere çalışır. İhtiyaç yoksa boş bırakılabilir.

Koşul: Döngü koşul sağlandığı sürece tekrar eder. Koşul sağlanıyorsa (true) döngüye girilir ve komutlar icra edilir. Döngü tamamlandığında koşulu tekrar kontrol eder. Koşul sağlanmıyorsa (false) döngüden çıkılır.

Artırım: Koşul her sağlandığında artırım işlemi de gerçekleştirilir.

Örnek: Yürüyen ışık programıdır.

```
byte i;
void setup()
{ for (i = 0; i < 10; i++)
  {
    pinMode(i, OUTPUT); // 0 - 9 pinleri çıkış yap.
  }
}
void loop()
{
  for (i = 0; i < 10; i++) // 0. pinden 9. pine kadar olan LED'leri sırayla...
  {
    digitalWrite(i, HIGH); // Yak.
    delay(200);
    digitalWrite(i, LOW); // Söndür.
  }
}
```

Örnek: İç içe kullanılan for döngüsünde ilk beş LED üç kez yanıp söner. Sonra son beş LED üç kez yanıp söner. Bu işlem on kere tekrar eder.

```
void setup() {
  byte i, x;
  int sure = 100;
  for (i = 0; i < 10; i++) {
    pinMode(i, OUTPUT); // 0 - 9 pinleri çıkış yap.
  }
  for (byte j = 0; j < 10; j++) { // Alttaki tüm kodları 10 kez çalıştır.
    for (x = 0; x < 3; x++) { // İçerdeki iki for döngüsünü üç kez tekrarla.
      for (i = 0; i < 5; i++) { // İlk beş LED'i yak. (Hepsi aynı anda yanar görünür.)
        digitalWrite(i, 1);
      }
      delay(sure);
      for (i = 0; i < 5; i++) { // İlk beş LED'i söndür.
        digitalWrite(i, 0);
      }
      delay(sure);
    }
    for (x = 0; x < 3; x++) { // İçerdeki iki for döngüsünü üç kez tekrarla.
      for (i = 5; i < 10; i++) { // Son beş LED'i yak.
        digitalWrite(i, 1);
      }
      delay(sure);
      for (i = 5; i < 10; i++) { // Son beş LED'i söndür.
        digitalWrite(i, 0);
      }
      delay(sure);
    }
  }
}
void loop() {}
```

Break

Normal döngü koşulunu atlayarak for, while veya do…while döngüsünden çıkmak için kullanılır. Ayrıca bir switch case ifadesinden çıkmak için de kullanılır.

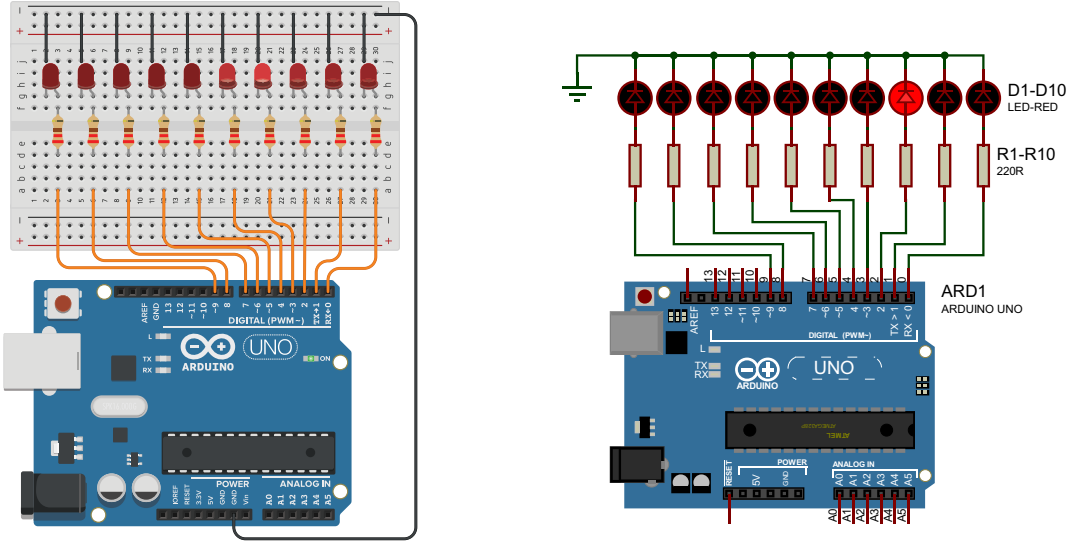
Örnek: Break kullanarak döngüden çıkar.

```
int esik = 40;
for (int x = 0; x < 255; x++) {
  analogWrite(PWMpin, x);
  deger = analogRead(sensorPin);
  if (deger > esik) {
    x = 0;
    break;
  }
  delay(50);}
```

Uygulama Adı: Kara Şimşek Uygulaması No.: 4

Amaç: **Kara şimşek uygulaması yapmak.**

Arduino'nun on adet çıkış pinine bağlanmış LED'i sırayla tek tek yakıp söndürmek için döngü komutları kullanılmıştır. Döngü komutları kullanılmazsa program tekrar eden komutlar nedeniyle çok uzun olacaktır. Uygulamada 0. pinden 9. pine kadar olan LED'ler sırayla yakılıp söndürülmektedir. Geriye dönüşte ise 8. pinden 1. pine kadar olan LED'ler sırayla yakılıp söndürülmektedir. "sure" değişkeninin içeriği değiştirilerek LED'lerin çalışma hızı ayarlanır.



Görsel 2.21: Kara şimşek uygulaması şema ve devresi

Kara şimşek uygulama programı aşağıdaki gibidir:

```
byte i;
int sure = 100;

void setup(){
  for (i = 0; i <= 9; i++){
    pinMode(i, OUTPUT); // 0 - 9 pinleri çıkış yap.
  }
}

void loop(){
  for (i = 0; i <= 9; i++){
    digitalWrite(i, HIGH);
    delay(sure);
    digitalWrite(i, LOW);
  }

 for (i = 8; i > 0; i--){ // İlk ve son LED dönüşlerde bir kez yanar.
    digitalWrite(i, HIGH);
```

```
  delay(sure);
    digitalWrite(i, LOW);
  }
}
```

DİKKAT

Arduino'nun 0. (Rx) ve 1. (Tx) pinleri USB kabloyla bilgisayara bağlıdır. Yükleme tuşuna basıldığında bu pinler kullanılır. 0. (Rx) pini devreye bağlıysa program karta yüklenirken hata verir. 0. pindeki **jumper** kablosu kaldırıldıktan sonra yükle düğmesine basılır. Program yüklendikten sonra **jumper** kablosu 0. pine tekrar bağlanır.

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.21'deki devreyi kurunuz.
4. Programı yazıp Arduino'ya yükleyerek devrenin çalışmasını gözlemleyiniz.
5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Uygulamayı for döngüsü kullanmadan yazınız.

```
int sure = 100;
void setup() {
  pinMode(0, OUTPUT);
  pinMode(1, OUTPUT);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
}
void loop() {
  digitalWrite(0, HIGH);
  delay(sure);
  digitalWrite(0, LOW);
  digitalWrite(1, HIGH);
  delay(sure);
  digitalWrite(1, LOW);
 digitalWrite(2, HIGH);
  delay(sure);
```

```
digitalWrite(2, LOW);

digitalWrite(3, HIGH);    delay(sure);
digitalWrite(3, LOW);
digitalWrite(4, HIGH);
delay(sure);
digitalWrite(4, LOW);
digitalWrite(5, HIGH);
delay(sure);
digitalWrite(5, LOW);
digitalWrite(6, HIGH);
delay(sure);
digitalWrite(6, LOW);
digitalWrite(7, HIGH);
delay(sure);
digitalWrite(7, LOW);
digitalWrite(8, HIGH);
delay(sure);
digitalWrite(8, LOW);
digitalWrite(9, HIGH);
delay(sure);
digitalWrite(9, LOW);
digitalWrite(8, HIGH);
delay(sure);
digitalWrite(8, LOW);
digitalWrite(7, HIGH);
delay(sure);
digitalWrite(7, LOW);
digitalWrite(6, HIGH);
delay(sure);
digitalWrite(6, LOW);
digitalWrite(5, HIGH);
delay(sure);
digitalWrite(5, LOW);
digitalWrite(4, HIGH);
delay(sure);
digitalWrite(4, LOW);
digitalWrite(3, HIGH);
delay(sure);
digitalWrite(3, LOW);
digitalWrite(2, HIGH);
delay(sure);
digitalWrite(2, LOW);
digitalWrite(1, HIGH);
delay(sure);
```

```
  digitalWrite(1, LOW);
  }
```

2. Uygulamayı for döngüsü kullanarak yirmi adet LED'le gerçekleştiriniz.

3. Aşağıdaki çakar uygulamasında ilk beş LED üç kez yanıp söner. Sonra son beş LED üç kez yanıp söner. Döngü sürekli tekrar eder.

```
byte i, x;
int sure = 100;
void setup() {
  for (i = 0; i < 10; i++) {
    pinMode(i, OUTPUT); // 0 - 9 pinleri çıkış yap.
  }
}
void loop() {
  for (x = 0; x < 3; x++) { // İçerdeki iki for döngüsünü üç kez tekrarla.
    for (i = 1; i < 6; i++) { // İlk beş LED›i yak. (Hepsi aynı anda yanar görünür.)
      digitalWrite(i, 1);
    }
    delay(sure);

    for (i = 1; i < 6; i++) { // İlk beş LED›i söndür.
      digitalWrite(i, 0);
    }
    delay(sure);
  }
  for (x = 0; x < 3; x++) { // İçerdeki iki for döngüsünü üç kez tekrarla.
    for (i = 6; i < 11; i++) { // Son beş LED'i yak.
      digitalWrite(i, 1);
    }
    delay(sure);

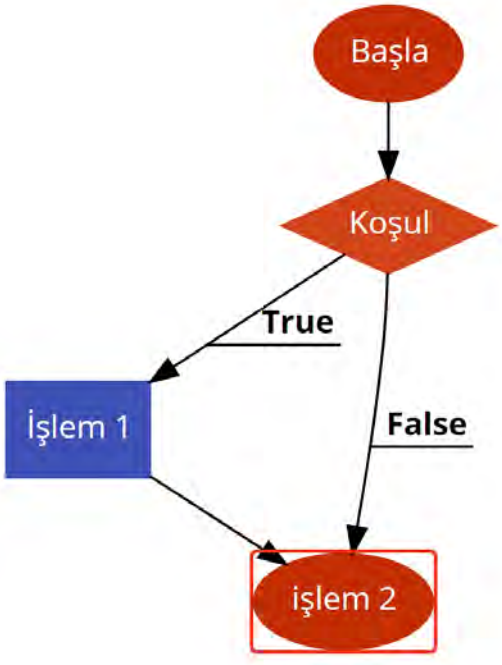
    for (i = 5; i < 11; i++) { // Son beş LED'i söndür.
      digitalWrite(i, 0);
    }
    delay(sure);
  }
}
```

Karar Yapıları

Koşula bağlı olarak iki veya daha fazla seçenekten birine dallanma işlemi gerçekleştiren komutlardır. Program şartlı olarak hangi işlemi yapacağına karar verir. Koşullar karşılaştırma operatörleriyle sınanır.

if (eğer) Yapısı

Koşula bağlı olarak tek bir işlemi yerine getiren yapıdır. Koşul ya doğrudur (true) ya da yanlıştır (false). Eğer koşul yanlışsa herhangi bir işlem yapılmaz. Koşul boolean bir ifadedir (Görsel 2.22).



Görsel 2.22: Koşula bağlı işlem

Örnek: if yapısının kullanım şekilleri gösterilmiştir.

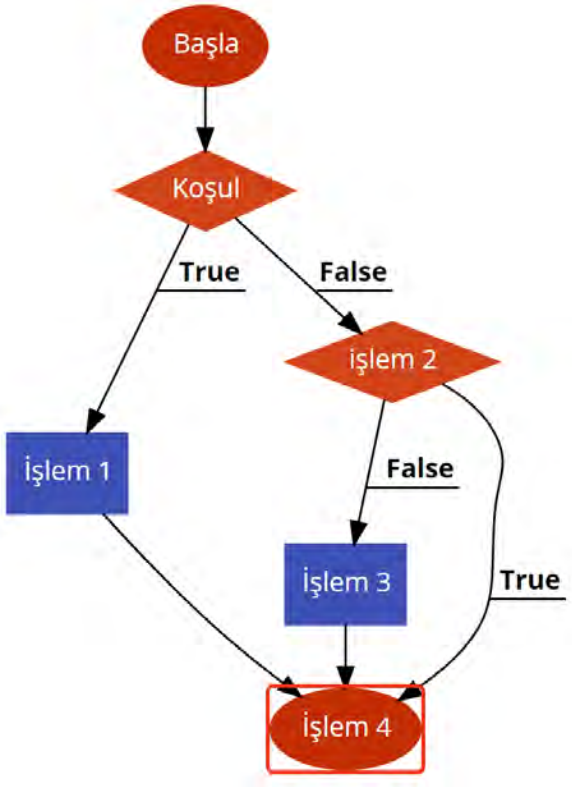
```
if (x > 120) digitalWrite(LED, HIGH);
if (x > 120)
digitalWrite(LEDpin, HIGH);

if (x > 120) {digitalWrite(LED, HIGH);}

if (x > 120) {
  digitalWrite(LED1, HIGH);
  digitalWrite(LED2, HIGH);
}
```

if-else Deyimi

Birden fazla koşulu test etmek için kullanılır. "else" komutuyla biten yapının sonunda koşul yoktur, varsayılan işlem vardır.



Görsel 2.23: Varsayılan koşula bağlı işlem

Görsel 2.23'te işlem 3 varsayılan işlem, işlem 4 koşul dışındaki koddur. Yapının sonunda "else" kullanılmazsa varsayılan işlem de kalkar.

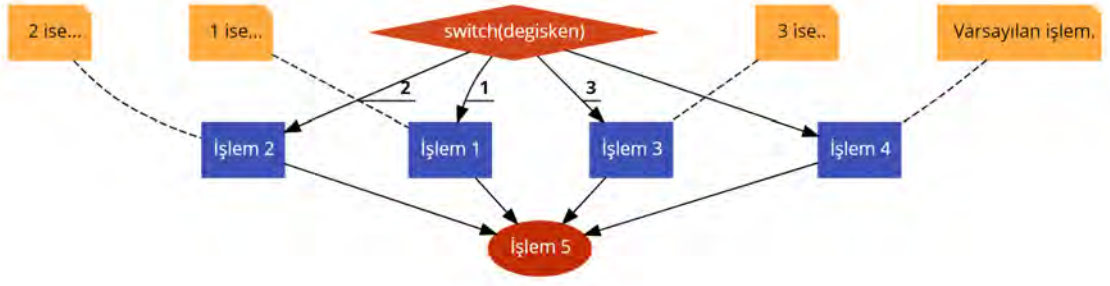
Örnek: if-else kullanımını gösterir.

```
if (sicaklik >= 70) {
  // Sistemi kapat.
}
else if (sicaklik >= 60) { // 60 <= sicaklik< 70
  // Dikkat riskli bölge.
}
else { // sicaklik< 60 (Koşul yok. Varsayılan işlem.)
  // Güvenli. Çalışmaya devam.
}
```

switch - case Deyimi

Sınanan değişkene göre işlemlerin seçiminde kullanılır. İç içe if-else yapısıyla benzerdir. Switch-case yapısı switch alanında belirtilen değişken değerini her bir case yapısındaki değerle karşılaştırır. Karşılaştırma sonucunda koşulu sağlanan case alanındaki kod çalıştırılır ve break komutuyla switch-case yapısından çıkılır. Hiçbir case koşulu sağlanmamışsa varsayılan (default) kodlar çalıştırılır.

Default kısmı isteğe bağlı olarak kullanılmayabilir (Görsel 2.24).



Görsel 2.24: Switch-case kullanımı

Örnek: switch-case kullanımını gösterir.

```
switch (degisken) {
case 1:
  // Değişken 1'e eşitse buradaki işlemler yapılır.
  break;
case 2:
  // Değişken 1'e eşitse buradaki işlemler yapılır.
  break;
default:
  // Hiçbir eşleşme olmazsa varsayılan işlemi yapar.
  // Varsayılan işlem tercihe bağlıdır.
  break;
}
```

NOT

Karar ve döngü işlemlerinde tek bir komut varsa { } (süslü parantezler) kullanılmasa da olur.

Diziler

Diziler indis numarasıyla ulaşılan değişken topluluğudur. Aşağıda dizi oluşturma örnekleri verilmiştir.

Örnek: Dizi tanımlamalarını gösterir.

```
int sayilar[6];
int pinler[] = {2, 4, 8, 3, 6};
int sensorDegerleri[6] = {2, 4, -8, 3, 2};
char mesaj[6] = "hello";
```

Diziler indis numarasıyla ulaşılan değişken topluluğudur. Aşağıda dizi oluşturma örnekleri verilmiştir.

Örnek: Dizi tanımlamalarını gösterir.

```
int sayilar[6];
int pinler[] = {2, 4, 8, 3, 6};
int sensorDegerleri[6] = {2, 4, -8, 3, 2};
char mesaj[6] = “hello”;
```

pnler[] dizisinde dizi uzunluğu (eleman sayısı) belirtilmemiştir. Derleyici elemanları sayarak uygun uzunluğu kendi oluşturur.

Dizilere Erişim

Diziler indis şeklindedir ve ilk indis 0'la başlar. Yukarıdaki örnekte sensorDegerleri[0] == 2'ye, sensorDegerleri[2] == -8'e eşittir. Aşağıdaki örnekte dizi elemanına değer atama görülmektedir.

```
sensorDegerleri[0] = 10 // Dizinin ilk elemanına 10 yüklenir.
x = sensorDegerleri[4] // x değişkenine 2 yüklenir.
```

Diziler for döngüsünde sıklıkla kullanılır. İçerik dizinin elemanlarına ulaşılacak şekilde yapılandırılır. Örnek programda ledler[] dizisinin elemanları LED'lerin bağlı olduğu pin numaralarını içermektedir. Arduino'nun 0. (Rx) ve 1. (Tx) pinleri yükleme esnasında “sök tak” yapmamak için tercih edilmemiştir.

Örnek: Dizi kullanılarak yapılmış kara şimşek programını gösterir.

```
byte i;
byte ledler[10] = {2, 3, 4, 5, 6, 7, 8, 9, 10, 11}; // On adet LED’in bağlandığı pinler.
int sure = 100;

void setup() {
  for (i = 2; i <= 11; i++)
    pinMode(i, OUTPUT); // 2 - 11 pinleri çıkış yap.
}

void loop()
{
  for (i = 0; i <= 9; i++)  {
    digitalWrite(ledler[i], HIGH);
    delay(sure);
    digitalWrite(ledler[i], LOW);
  }
  for (i = 8; i > 0; i--) { // İlk ve son LED dönüşlerde bir kez yanar.
    digitalWrite(ledler[i], HIGH);
    delay(sure);
    digitalWrite(ledler[i], LOW);
  }
}
```

Çok Boyutlu Diziler

Her elemanı bir dizi olan dizilere **çok boyutlu dizi** denir. İki boyutlu dizilere **matris** denir. Matrisler satır ve sütunlardan oluşan tablolar gibidir.

Örnek: Çok boyutlu dizi tanımlamalarını gösterir.

```
int dizi[3][4] = {{10, 5, 12, 20}, {7, 3, 14, 22}, {8, 7, 8, 24}};

// Dizinin her elemanına atama yapıldı.

int matris[4][3] = {{25, 33, 12}, {87, 66, 63}, {45, 90, 34}, {22, 46, 14}};

int x = dizi[0][3]; // x'e 10 sayısı yüklendi.

int y = matris[3][0]; // y'ye 22 sayısı yüklendi.
```

Örnek: Matrisle yedi segment display uygulamasını gösterir.

```
const byte a = 6, b = 5, c = 4, d = 3, e = 2, f = 1, g = 0 ; // Segmentlerin bağlandığı pinler.
byte rakamSegmentleri[10][7] = { // Her rakam için segment durumları.

  // g, f, e, d, c, b, a
  { 0, 1, 1, 1, 1, 1, 1 }, // = 0
  { 0, 0, 0, 0, 1, 1, 0 }, // = 1
  { 1, 0, 1, 1, 0, 1, 1 }, // = 2
  { 1, 0, 0, 1, 1, 1, 1 }, // = 3
  { 1, 1, 0, 0, 1, 1, 0 }, // = 4
  { 1, 1, 0, 1, 1, 0, 1 }, // = 5
  { 1, 1, 1, 1, 1, 0, 1 }, // = 6
  { 0, 0, 0, 0, 1, 1, 1 }, // = 7
  { 1, 1, 1, 1, 1, 1, 1 }, // = 8
  { 1, 1, 0, 1, 1, 1, 1 }  // = 9
};

void setup() {
  for (byte i = 0; i <= 6; i++) { // Yedi adet pin çıkış olarak ayarlandı.
    pinMode(i, OUTPUT); // 0 - 6 pinleri çıkış yap.
  }
}

void loop() {

for (byte rakam = 0; rakam <= 9; rakam++) {
for (byte segment = 0; segment < 7; segment++) {
      digitalWrite(segment , rakamSegmentleri[rakam][segment]);
    }
    delay(200);
  }
}
```

Fonksiyonlar

Program içinde pek çok yerde tekrar eden kod bloku varsa okunabilirliği arttırmak için bu kod bloku fonksiyon olarak diğer fonksiyonların dışında tanımlanır ve istenilen yerde bu fonksiyon çağrılarak kod bloku çalıştırılır. Dört çeşit fonksiyon tanımlaması mevcuttur.

Değer Almayan ve Değer Döndürmeyen Fonksiyonlar

Bu tip fonksiyonların başına void ifadesi yazılır. Değer almayan fonksiyonların parantezlerinin içi boştur. Aşağıda blink (yanıp sönen LED) uygulamasının fonksiyona dönüştürülmüş şekli verilmiştir.

Örnek: Değer almayan ve değer döndürmeyen blink() fonksiyonunu gösterir.

```
const byte LED = 13;
void setup() {
  pinMode(LED, OUTPUT);
}
void loop() {
  blink(); // Blink fonksiyonunu çağır.
}
void blink() { // Değer almayan ve değer döndürmeyen.
  digitalWrite(LED, HIGH);
  delay(1000);
  digitalWrite(LED, LOW);
  delay(1000);
}
```

Değer Alan ve Değer Döndürmeyen Fonksiyonlar

Bu tip fonksiyonların başına void ifadesi yazılır. Fonksiyon parantezlerinin içine belirtilen değişken tipinde veri girilir.

Örnek: Değer alan ve değer döndürmeyen blink() fonksiyonunu gösterir.

```
const byte LED = 13;
void setup() {
 pinMode(LED, OUTPUT);
}
void loop() {
  blink(500); // 500 değerini almış blink foksiyonunu çağır.
}
void blink(int sure) { // int tipinde değer alan, değer döndürmeyen.
  digitalWrite(LED, HIGH);
  delay(sure);
  digitalWrite(LED, LOW);
  delay(sure);
}
```

Örnek: Yedi segment display uygulamasıdır.

```
const byte a = 6, b = 5, c = 4, d = 3, e = 2, f = 1, g = 0;
// Segmentlerin bağlandığı pinler.
int sure = 200;

void setup() {
  for (byte i = 0; i <= 6; i++) {
    pinMode(i, OUTPUT); // 0 - 6 pinleri çıkış yap.
  }
}

void loop() {
  for (byte rakam = 0; rakam <= 9; rakam++) { // Rakamları ileri say.
    yazdir(rakam); // Girilen rakamı yazdir() fonksiyonuna gönder.
    delay(sure);
  }
  for (byte rakam = 8; rakam >= 1; rakam--) { // Rakamları geri say.
    yazdir(rakam);
    delay(sure);
  }
}

void yazdir(byte rakam) { // byte tipinde değer alan, değer döndürmeyen.
  switch (rakam) { // Rakam değişkenini seçeneklerden bul.
    case 0 :
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, HIGH);
      digitalWrite(e, HIGH);
      digitalWrite(f, HIGH);
      digitalWrite(g, LOW);
      break;
    case 1:
      digitalWrite(a, LOW);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, LOW);
      digitalWrite(e, LOW);
      digitalWrite(f, LOW);
      digitalWrite(g, LOW);
      break;
    case 2:
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, LOW);
```

```
    digitalWrite(d, HIGH);
    digitalWrite(e, HIGH);
    digitalWrite(f, LOW);
    digitalWrite(g, HIGH);
    break;
  case 3:
    digitalWrite(a, HIGH);
    digitalWrite(b, HIGH);
    digitalWrite(c, HIGH);
    digitalWrite(d, HIGH);
    digitalWrite(e, LOW);
    digitalWrite(f, LOW);
    digitalWrite(g, HIGH);
    break;
  case 4:
    digitalWrite(a, LOW);
    digitalWrite(b, HIGH);
    digitalWrite(c, HIGH);
    digitalWrite(d, LOW);
    digitalWrite(e, LOW);
    digitalWrite(f, HIGH);
    digitalWrite(g, HIGH);
    break;
  case 5:
    digitalWrite(a, HIGH);
    digitalWrite(b, LOW);
    digitalWrite(c, HIGH);
    digitalWrite(d, HIGH);
    digitalWrite(e, LOW);
    digitalWrite(f, HIGH);
    digitalWrite(g, HIGH);
    break;
  case 6:
    digitalWrite(a, HIGH);
    digitalWrite(b, LOW);
    digitalWrite(c, HIGH);
    digitalWrite(d, HIGH);
    digitalWrite(e, HIGH);
    digitalWrite(f, HIGH);
    digitalWrite(g, HIGH);
    break;
  case 7:
    digitalWrite(a, HIGH);
    digitalWrite(b, HIGH);
```

```
      digitalWrite(c, HIGH); digitalWrite(LED, HIGH);
  delay(1000);
  digitalWrite(LED, LOW);
  delay(1000);
  int saniye = millis() / 1000; // Programın milisaniye cinsinden çalışma süresini 1000'e böl.
 return saniye; // saniye değişkenini döndür.
}
      digitalWrite(d, LOW);
      digitalWrite(e, LOW);
      digitalWrite(f, LOW);
      digitalWrite(g, LOW);
      break;
    case 8:
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
      digitalWrite(d, HIGH);
      digitalWrite(e, HIGH);
      digitalWrite(f, HIGH);
      digitalWrite(g, HIGH);
      break;
    case 9:
      digitalWrite(a, HIGH);
      digitalWrite(b, HIGH);
      digitalWrite(c, HIGH);
```


Değer Almayan ve Değer Döndüren Fonksiyonlar

Fonksiyonun değer döndürmesi (return) bir çıktı oluşturduğunu gösterir. Değer döndüren fonksiyonlar bir değişkene atanabilir. Değer döndüren fonksiyonların başına döndürülen verinin değişken tipi yazılır. Değer almadığı için fonksiyon parantezlerinin içi boştur.

Örnek: Değer almayan ve değer döndüren blink() fonksiyonunu gösterir.

```
const byte LED = 13;
void setup() {
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  blink(); // blink fonksiyonunu çağır.
  int zaman = blink(); // Değer döndüren fonksiyonlar bir değişkene atanabilir.
  Serial.print(zaman);
  Serial.println(" saniyedir program çalışmaktadır.");
}
int blink() { // Değer almayan ancak int tipinde değer döndüren.
```

```
      digitalWrite(d, HIGH);
      digitalWrite(e, LOW);
      digitalWrite(f, HIGH);
      digitalWrite(g, HIGH);
      break;
  }
}
```

Değer Alan ve Değer Döndüren Fonksiyonlar

Bu tip fonksiyonların başına döndürülen verinin değişken tipi yazılır. Fonksiyon parantezlerinin içine belirtilen değişken tipinde veri girilir.

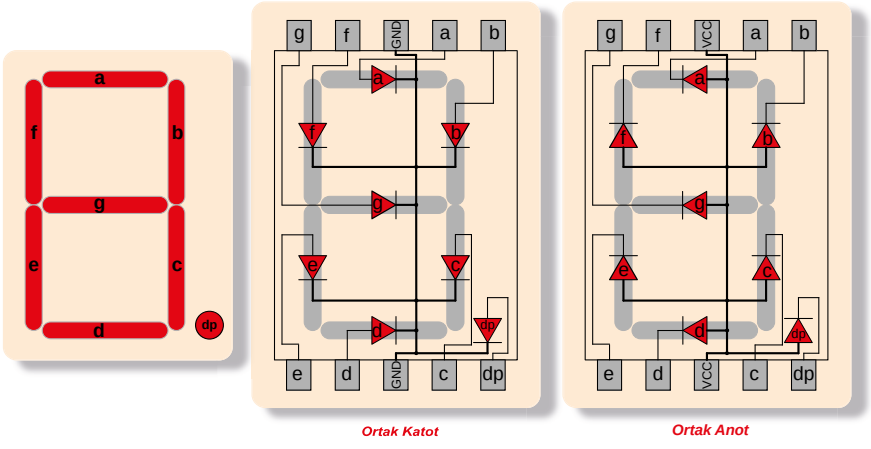
Örnek: Değer alan ve değer döndüren blink() fonksiyonunu gösterir.

```
const byte LED = 13;
void setup() {
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  blink(100); // 100 değerini almış blink foksiyonunu çağır.
  Serial.print(blink(100)); // Blink fonksiyonundan dönen değeri yazdır.
  Serial.println(" saniyedir program çalışmaktadır.");
}
int blink(int sure) {
  digitalWrite(LED, HIGH);
  delay(sure);
  digitalWrite(LED, LOW);
  delay(sure);
  return millis() / 1000; // millis() / 1000 işleminin sonucunu döndür.
}
```

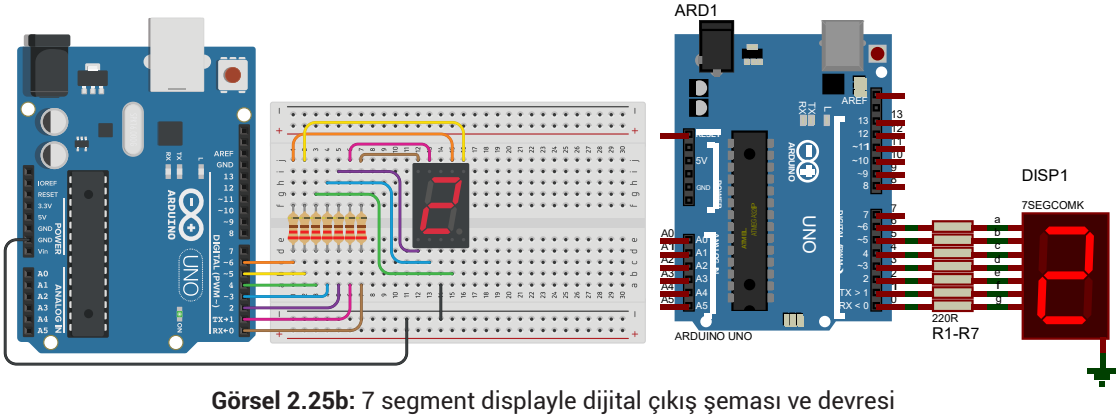
Uygulama Adı: Segment Displayle Dijital Çıkış Uygulaması No.: 5

Amaç: 7 segment displayle dijital çıkış uygulaması yapmak.

Görsel 2.25a'da ortak katotlu ve ortak anotlu 7 segment displayin yapısı, Görsel 2.25b'de ise dijital çıkış şeması ve devresi görülmektedir. Displayin a, b, c, d, e, f, g segmentleri sırasıyla Arduino'nun 6, 5, 4, 3, 2, 1, 0 numaralı pinlerine bağlıdır. Ortak katot Arduino'nun GND ucuna bağlıdır. Karar yapısı olarak switch case kullanılmıştır. Rakam değişkenin içeriğine göre ilgili komut kümesi çalıştırılarak dijital çıkışların durumuna göre rakamlar displayde yanmaktadır. For döngüsüyle 0,2 saniyede bir rakamlar artırılmaktadır. Tablo 2.8'de ortak katot display giriş kodları verilmiştir.



Görsel 2.25a: Ortak katotlu ve ortak anotlu 7 segment displayin yapısı



Görsel 2.25b: 7 segment displayle dijital çıkış şeması ve devresi

Tablo 2.8: Ortak Katot Display Giriş Kodları

Pin	6	5	4	3	2	1	0
Rakam	a	b	c	d	e	f	g
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1

7 segment displayle dijital çıkış uygulama programı aşağıdaki gibidir:

```
const byte a = 6, b = 5, c = 4, d = 3, e = 2, f = 1, g = 0;
void setup() {
  for (byte i = 0; i <= 6; i++)
  {
    pinMode(i, OUTPUT); // 0 - 6 pinleri çıkış yap.
  }
}
void loop() {
  for (byte rakam = 0; rakam <= 9; rakam++) { // Rakamları yukarı say.
    switch (rakam) { // Rakam değişkenini seçeneklerden bul.
      case 0 :
        digitalWrite(a, HIGH);
        digitalWrite(b, HIGH);
        digitalWrite(c, HIGH);
        digitalWrite(d, HIGH);
        digitalWrite(e, HIGH);
        digitalWrite(f, HIGH);
        digitalWrite(g, LOW);
        break;
      case 1:
        digitalWrite(a, LOW);
        digitalWrite(b, HIGH);
        digitalWrite(c, HIGH);
   digitalWrite(d, LOW);
        digitalWrite(e, LOW);
        digitalWrite(f, LOW);
        digitalWrite(g, LOW);
        break;
```

```
case 2:
       digitalWrite(a, HIGH);
  digitalWrite(b, HIGH);
  digitalWrite(c, LOW);
  digitalWrite(d, HIGH);
  digitalWrite(e, HIGH);
  digitalWrite(f, LOW);
  digitalWrite(g, HIGH);
  break;
case 3:
  digitalWrite(a, HIGH);
  digitalWrite(b, HIGH);
  digitalWrite(c, HIGH);
  digitalWrite(d, HIGH);
  digitalWrite(e, LOW);
  digitalWrite(f, LOW);
  digitalWrite(g, HIGH);
  break;
case 4:
  digitalWrite(a, LOW);
  digitalWrite(b, HIGH);
  digitalWrite(c, HIGH);
  digitalWrite(d, LOW);
  digitalWrite(e, LOW);
  digitalWrite(f, HIGH);
  digitalWrite(g, HIGH);
  break;
case 5:
  digitalWrite(a, HIGH);
  digitalWrite(b, LOW);
  digitalWrite(c, HIGH);
  digitalWrite(d, HIGH);
  digitalWrite(e, LOW);
  digitalWrite(f, HIGH);
  digitalWrite(g, HIGH);
  break;
case 6:
   digitalWrite(a, HIGH);
  digitalWrite(b, LOW);
  digitalWrite(c, HIGH);
  digitalWrite(d, HIGH);
  digitalWrite(e, HIGH);
  digitalWrite(f, HIGH);
  digitalWrite(g, HIGH);
  break;
```

```
                    case 7:
        digitalWrite(a, HIGH);
        digitalWrite(b, HIGH);
        digitalWrite(c, HIGH);
        digitalWrite(d, LOW);
        digitalWrite(e, LOW);
        digitalWrite(f, LOW);
        digitalWrite(g, LOW);
        break;
      case 8:
        digitalWrite(a, HIGH);
        digitalWrite(b, HIGH);
        digitalWrite(c, HIGH);
        digitalWrite(d, HIGH);
        digitalWrite(e, HIGH);
        digitalWrite(f, HIGH);
        digitalWrite(g, HIGH);
        break;
      case 9:
        digitalWrite(a, HIGH);
        digitalWrite(b, HIGH);
        digitalWrite(c, HIGH);
        digitalWrite(d, HIGH);
        digitalWrite(e, LOW);
        digitalWrite(f, HIGH);
        digitalWrite(g, HIGH);
        break;
    } // switch case bitti.
  } // for döngüsü bitti.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.25'teki devreyi kurunuz.
4. Programı yazıp Arduino'ya yükleyiniz. Displayin çalışmasını gözlemleyiniz.
5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

Bir programı yazmanın pek çok yolu vardır. Bu uygulamanın programı switch case, iki boyutlu dizi vb. kullanılarak yazılabilir. Bunlardan biri olan port komutlarıyla yapılmış programı deneyiniz.

Aşağıdaki Arduino kodunda DDRD port registerla portları giriş çıkış olarak tanımlama yapılmıştır. DDRD = B01111111; komutuyla port register'a 8 bitlik binary sayı atayarak 0 ile 7. port giriş, 1 ile 0-6. portlar çıkış olarak ayarlanmıştır.

PORTD = B01111110; komutuyla displaye 0 sayısı yazdırılır. Bu komutta a, b, c, d, e, f segmentlerine (6, 5, 4, 3, 2, 1 pinleri) 1 bilgisi gönderilirken g segmentine (0. Pin) 0 gönderilmiştir. Programda 0, 1, 2 sayıları yazdırılmıştır. 3, 4, 5, 6, 7, 8, 9 sayılarını programa ekleyerek çalışmasını gözlemleyiniz.

```
void setup() {
DDRD = B01111111; // 0-6 numaralı pinler çıkış, 7 numaralı pin giriş olarak ayarlandı.
}
void loop() {
PORTD = B01111110; // 0-5 numaralı pinlere 1, 6. pine 0 bilgisi gönderildi. 0 rakamı yazdırıldı.
delay(1000);
PORTD = B00110000; // 1 rakamını yaz.
delay(1000);}
PORTD = B01101101; // 2 rakamını yaz.
delay(1000);
}
```

Sorular

1. pinMode() fonksiyonuyla tek bir pin giriş veya çıkış olarak ayarlanırken DDRD registerla kaç pin giriş veya çıkış olarak ayarlanmaktadır? Belirtiniz.
2. Display ortak anotlu olsaydı PORTD = B00000001; komutuyla displayde hangi sayı görünürdü? Açıklayınız.
3. 7 segment display uygulamasını matris kullanarak yazınız.

2.7.3. Dijital Giriş İşlemleri

digitalRead() Fonksiyonu

pinMode() fonksiyonuyla INPUT olarak ayarlanan pinin HIGH (5 V) veya LOW (0 V) durumunu okur. Çıktı (dönüş) olarak HIGH veya LOW (1 veya 0) olma durumunu döndürür.

digitalRead(pin)

pin: Pin numarası.

Örnek: Dijital girişin 5 V veya 0 V'ta olma durumunu okur.

```
void setup() {
pinMode(2,INPUT); // 2 numaralı pin giriş.
pinMode(13,OUTPUT); // 13 numaralı pin çıkış.
}
void loop() {
digitalWrite(13,digitalRead(2)); // 2 numaralı pini oku. Gelen 0 veya 1 bilgisini
// 13 numaralı pine gönder.
}
```

Örnek: Tüm dijital çıkışlı sensörler için temel programdır.

```
const byte giris = 2; // Giriş pini.
const byte cikis = 13; // Çıkış pini.


void setup() {
  pinMode(giris, INPUT_PULLUP); // Çıkışı aktif “0” olan sensörler için.
  // pinMode(giris, INPUT); // Çıkışı aktif “1” olan sensörler için kullanılır.
  pinMode(cikis, OUTPUT);
}

void loop() {
  bool girisDurum = digitalRead(giris); // Giriş bilgisini oku.
  if (girisDurum == 0) { // Sensör aktifse yapılacaklar...
    digitalWrite(cikis, HIGH);
  }
  else { // Sensör pasifse yapılacaklar...
    digitalWrite(cikis, LOW);
  }
}
```

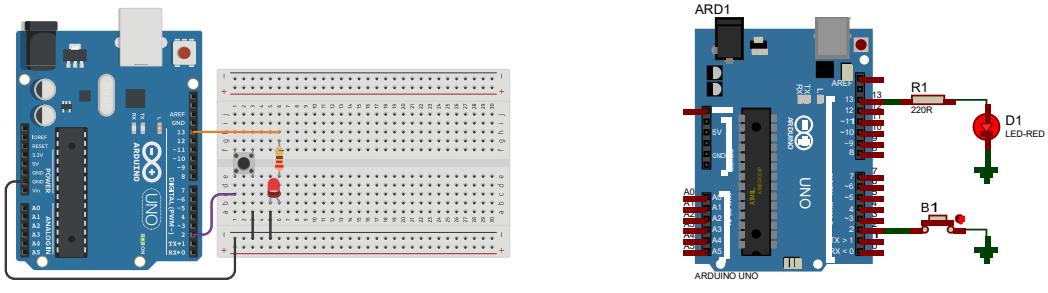
Uygulama Adı: Butonla Dijital Giriş Uygulaması No.: 6

Amaç: Butonla dijital giriş uygulaması yapmak.

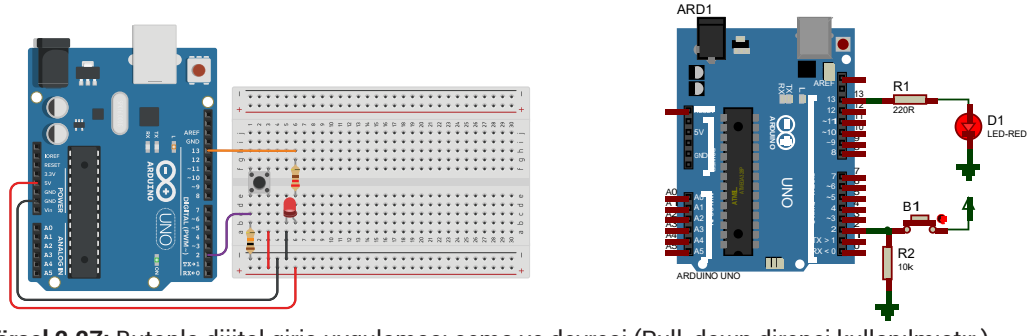
Görsel 2.26'da dâhilî pull-up direnci pinMode(buton, INPUT_PULLUP); komutuyla aktif edilmiştir. Butonla dijital giriş uygulanmıştır. Görsel 2.26'da dâhilî pull-up direncinden dolayı butona basılmadığı sürece 2 numaralı giriş pininde lojik 1 bilgisi mevcuttur. Butona basıldığında 2 numaralı giriş pinine GND'den lojik 0 bilgisi verilir.

Pull-down direnci kullanılacaksa buton ve direnç Görsel 2.27'deki şekilde bağlanır.10 kΩ'luk pull-down direncinden dolayı butona basılmadığı sürece 2 numaralı giriş pininde lojik 0 bilgisi mevcuttur. Butona basıldığında 2 numaralı giriş pinine 5 V beslemeden lojik 1 bilgisi verilir.

Booelan tipinde tanımlanan butonDurum değişkeni sadece 0 (false) veya 1 (true) değerini alır. if karar yapısıyla butonun durumu kontrol edilir. Buton basılıyken GND'den 2 numaralı girişe 0 bilgisi gelir. Bu durumda LED yakılır. Butondan el çekildiğinde INPUT_PULLUP'tan dolayı butonDurum değişkeni 1 olur ve LED söner.



Görsel 2.26: Butonla dijital giriş uygulaması şema ve devresi (Dâhilî pull-up direnci kullanılmıştır.)



Görsel 2.27: Butonla dijital giriş uygulaması şema ve devresi (Pull-down direnci kullanılmıştır.)

Butonla dijital giriş uygulama programı aşağıdaki gibidir:

```
const byte LED = 13; // 13 numaralı pini LED değişkenine ata.
const byte buton = 2; // 2 numaralı pini buton değişkenine ata.
boolean butonDurum; // butonDurum adında boolean değişken oluştur.
void setup() {
  pinMode(buton, INPUT_PULLUP); // 2 numaralı girişteki pull-up direncini aktif yap.
  pinMode(LED, OUTPUT); // LED'in bağlı olduğu pini çıkış olarak ayarla.
}
void loop() {
  butonDurum = digitalRead(buton); // Butona basıldı (0), basılmadı (1) bilgisini oku.
  if (butonDurum == 0) // Buton basılıysa...
    digitalWrite(LED, HIGH); // LED'i yak.
    else digitalWrite(LED, LOW); // LED'i söndür.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.26'daki devreyi kurunuz.
4. Programı yazıp Arduino'ya yükleyiniz.
5. Butona basıp bırakarak LED'in çalışmasını gözlemleyiniz.
6. Görsel 2.27'deki devreyi kurunuz.
7. Programı pull-down direncine göre düzenleyerek Arduino'ya yükleyiniz.
8. Butona basıp bırakarak LED'in çalışmasını gözlemleyiniz.
9. Programı hiç değişken kullanmadan tekrar yazınız.
10. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

Aşağıdaki programla Arduino'ya yüklenip çalışması gözlemlenen program aynı işi görmektemi - dir? Aralarındaki farklar nelerdir? Açıklayınız.

```
void setup() {
pinMode(2,INPUT_PULLUP);
pinMode(13,OUTPUT);
}
void loop() {
digitalWrite(13,!digitalRead(2));
}
```

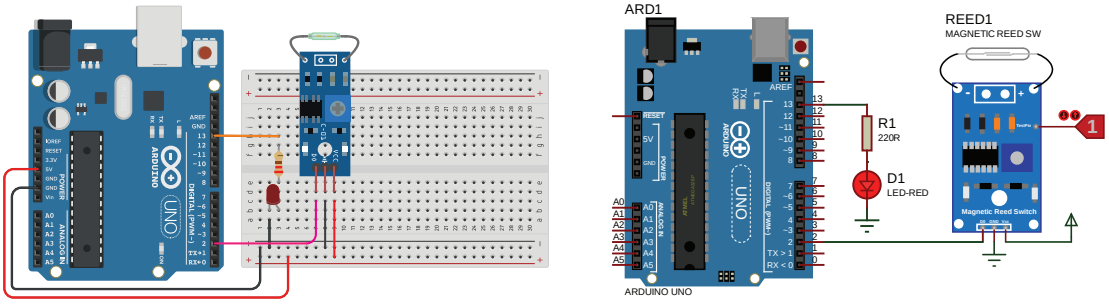
Soru

1. Arduino'da haricî pull-up direncine neden ihtiyaç yoktur? Açıklayınız.

Uygulama Adı: Reed Switch'le Dijital Giriş Uygulaması No.: 7

Amaç: Reed switch'le dijital giriş uygulaması yapmak.

Görsel 2.28'de kullanılan reed röle modülüne mıknatıs yaklaştırıldığında veya manyetik alan içine girdiğinde D0 çıkışı lojik 1 olmaktadır. Modül üzerinde besleme (5 V), GND ve D0 dijital çıkış pini yer almaktadır. Mıknatıs yaklaştırıldığında D0 pininden 5 V (1) çıkış vermektedir ve aynı zamanda üzerindeki LED de yanmaktadır. Bazı modüllerde A0 analog çıkış pini de bulunmaktadır.



Görsel 2.28: Reed switch'le dijital giriş şeması ve devresi

Reed switch'le dijital giriş uygulama programı aşağıdaki gibidir:

```
const byte LED = 13; // 13 numaralı pini LED değişkenine ata.
const byte reedRole = 2; // 2 numaralı pini reedRole değişkenine ata.
void setup() {
pinMode(reedRole, INPUT); // 2 numaralı pini giriş olarak ayarla.
pinMode(LED, OUTPUT); // LED'in bağlı olduğu pini çıkış olarak ayarla.
}
void loop() {
boolean reedRoleDurum = digitalRead(reedRole);
/* Değişkenler fonkisyon içinde de tanımlanabilir.
Ancak sadece tanımlandığı fonksiyon içinde kullanılabilir.
Mıknatıs yaklaştı (1), Mıknatıs uzaklaştı (0) bilgisini oku.*/
if (reedRoleDurum == 1) // reedRoleDurum değişkeni içindeki bilgi 1'se...
digitalWrite(LED, reedRoleDurum); // reedRoleDurum değişkenindeki bilgiyi (1) LED'e yaz.
else // reedRoleDurum değişkeni içindeki bilgi 0'sa...
digitalWrite(LED, reedRoleDurum); // reedRoleDurum değişkenindeki bilgiyi (0) LED'e yaz.
// if kontrolünden sonra tek bir komut varsa { } süslü parantezler kullanılmasa da olur.
}
```

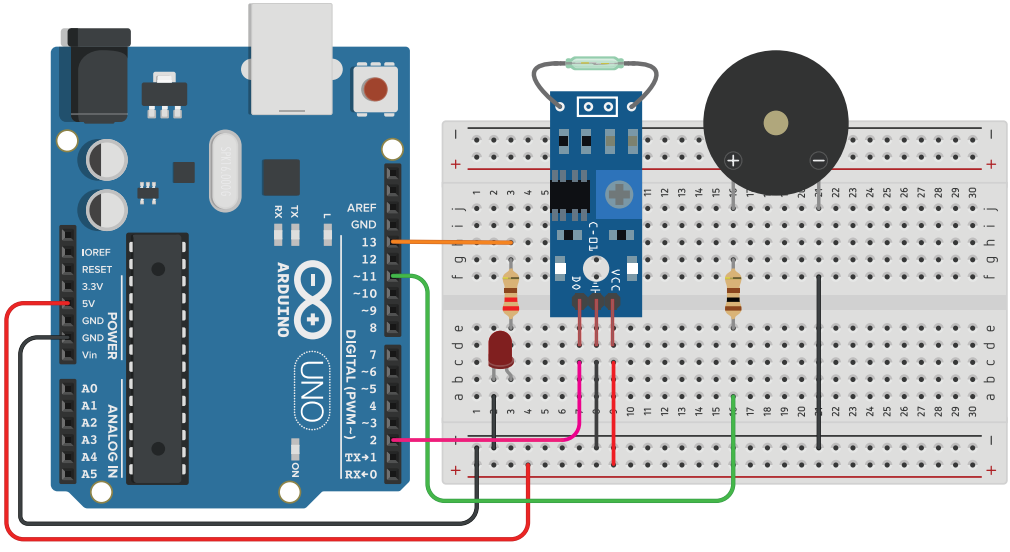
İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.28'deki devreyi kurunuz.

4. Programı yazıp Arduino'ya yükleyiniz.
5. Mıknatısı reed röleye yaklaştırınız LED'in yandığını gözlemleyiniz.
6. Mıknatısı reed röleden uzaklaştırınız. LED'in söndüğünü gözlemleyiniz.
7. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Mıknatıs yaklaştığında LED sönecek şekilde yazılımı düzenleyiniz.
2. Mıknatıs yaklaştığında sesli ikaz verecek donanımsal değişikliği yapınız.



Sorular

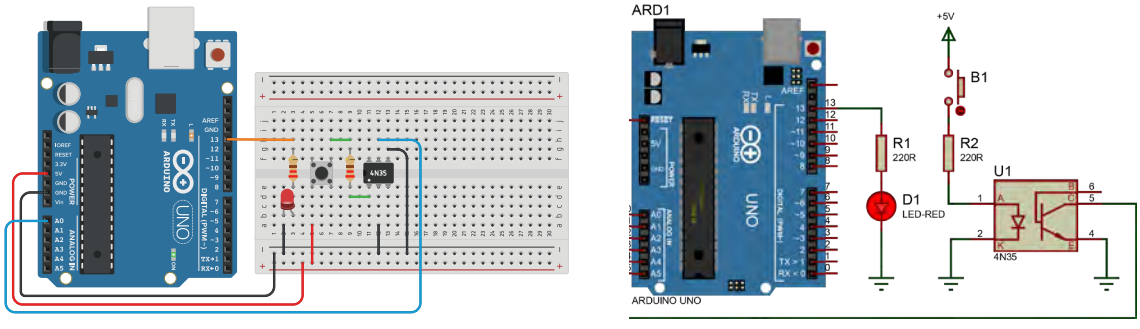
1. Program da if(1) ise LED'in durumu nedir? Açıklayınız.
2. Program da if(0) ise LED'in durumu nedir? Açıklayınız.
3. if(x) ifadesinde x her zaman boolean bir ifade midir? Belirtiniz.
4. Program if yapısı kullanılmadan yazılabilir mi? Açıklayınız.

Uygulama Adı: Optokuplörle dijital giriş uygulaması No.: 8

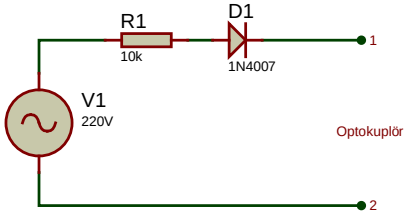
Amaç: Optokuplörle dijital giriş uygulaması yapmak.

Görsel 2.29.a'daki devrede optokuplör A0'a (14 numaralı pin) bağlanmıştır. Bu uygulamada A0 pini dijital giriş olarak kullanılmıştır. Optokuplör iletime geçtiğinde normalde pull-up'tan dolayı 1 olan A0 C-E üzerinden GND'ye bağlanır (0 olur.). Programda üç adet boolean (bool) değişken yan yana tanımlanmış ve ikisine değer atanmıştır. Programda HIGH veya 1 kullanılan yerlerde içinde 1 bilgisi olan **yak** değişkeni kullanılmıştır.

Optokuplörün 1 ve 2 numaralı uçlarına Görsel 2.29.b'deki devre bağlanarak 220 V gerilim Arduino'ya zarar vermeden test edilebilir.



Görsel 2.29a: Optokuplörle dijital giriş şeması ve devresi



Görsel 2.29b: Optokuplör ve Arduino ile 220 V kontrol

Optokuplörle dijital giriş uygulama programı aşağıdaki gibidir:

```
const byte LED = 13; // 13 numaralı pini LED değişkenine ata.
const byte opto = 14; // 14 numaralı pin A0 pinidir.
bool optoDurum, yak = 1, sondur = 0;
void setup() {
pinMode(opto, INPUT_PULLUP); // 14 numaralı pin 5 V'a (1) çekildi.
pinMode(LED, OUTPUT); // LED'in bağlı olduğu pini çıkış olarak ayarla.
}
void loop() {
optoDurum = digitalRead(opto);
if (optoDurum == 0) // optoDurum 0'a eşitse..
digitalWrite(LED, yak); // 13 numaralı pine 1 gönder.
else
digitalWrite(LED, sondur); // 13 numaralı pine 0 gönder.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.29a'daki devreyi kurunuz.
4. Programı yazıp Arduino'ya yükleyiniz.
5. Butona bastığınızda LED'in yandığı gözlemleyiniz.
6. Görsel 2.29b'deki **devreyi test etmek için öğretmeninizden yardım alınız.**
7. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Programı değişken ve if kullanmadan yazınız (Butonla dijital giriş uygulamasına bakınız.).

Sorular

1. Yüksek gerilimde oluşacak bir aksaklıkta Arduino kartını koruyan eleman hangisidir? Belirtiniz.
2. Aşağıdaki program bu uygulamadaki programla aynı işlevi görür mü?

```
const byte giris = 2; // Giriş pini.
const byte cikis = 13; // Çıkış pini.


void setup() {
  pinMode(giris, INPUT_PULLUP); // Çıkışı aktif “0” olan sensörler için.
  pinMode(cikis, OUTPUT);
}

void loop() {
  bool girisDurum = digitalRead(giris); // Giriş bilgisini oku.
  if (girisDurum == 0) { // Sensör aktifse yapılacaklar...
    digitalWrite(cikis, HIGH);
  }
  else { // Sensör pasifse yapılacaklar...
    digitalWrite(cikis, LOW);
  }
}
```

2.7.4. Zaman Gecikmesi Komutları

Arduino kodları içerisinde farklı işlevleri olan program durdurma komutları vardır.

delay() Fonksiyonu

Programı, parametre olarak belirtilen süre kadar (milisaniye cinsinden) duraklatır (1000 milisaniye 1 saniyedir.). delay fonksiyonu işlemleri durduğu için süre sonuna kadar diğer işlemler (sensör okuma vb.) yapılmayacaktır. Gelişmiş bir programda delay() yerine milis() fonksiyonu kullanılır.

delay(ms)

ms: Duraklatılacak milisaniye sayısı. Veri tipi: unsigned long.

delayMicroseconds() Fonksiyonu

Programı, parametre olarak belirtilen süre kadar (mikrosaniye cinsinden) duraklatır (1000000 mikrosaniye 1 saniyedir.).

delayMicroseconds(us)

us: Duraklatılacak mikrosaniye sayısı. Veri tipi: unsigned int.

Örnek: 12 numaralı pinden 10 mikrosaniyelik kare dalga gönderilir.

```
void setup(){
pinMode(12,OUTPUT);
}
void loop(){ // 12 numaralı pinden 10 mikrosaniyelik kare dalga gönderiliyor.
digitalWrite(12,0);
delayMicroseconds(10);
digitalWrite(12,1);
delayMicroseconds(10);
}
```

millis() Fonksiyonu

Arduino kartının mevcut programı çalıştırmaya başlamasından itibaren geçen milisaniye sayısını döndürür. Yaklaşık 50 gün sonra bu sayı sıfıra dönerek tekrar eder.

zaman = millis();

Giriş parametresi yoktur. Unsigned long veri tipinde geri döndürdüğü sayı bir değişkene atanarak kullanılabilir.

Örnek: Program başladıktan sonra geçen zamanı milisaniye cinsinden ekrana yazar.

```
unsigned long gecenZaman;
void setup() {
  Serial.begin(9600);
}
void loop() {
 Serial.print("Geçen zaman: "); // Arduino'daki reset butonunuyla zaman sıfırlanır.
```

```
  gecenZaman = millis();
 Serial.println(gecenZaman);// Program başladıktan sonra geçen zamanı (ms) ekrana yaz.
  delay(1000); // 1 saniye bekle. (Saniyede bir ekrana yazdır.)
```

micros() Fonksiyonu

Arduino kartının mevcut programı çalıştırmaya başlamasından itibaren geçen mikrosaniye sayısını döndürür. Yaklaşık 70 dakika sonra bu sayı sıfıra dönerek tekrar eder.

zaman = micros();

Giriş parametresi yoktur. Unsigned long veri tipinde geri döndürdüğü sayı bir değişkene atanarak kullanılabilir.

Örnek: Program başladıktan sonra geçen zamanı mikrosaniye cinsinden ekrana yazar.

```
unsigned long gecenZaman;
void setup() {
  Serial.begin(9600);
}
void loop() {
 Serial.print("Geçen zaman: ");// Arduino'daki reset butonunuyla zaman sıfırlanır.
  gecenZaman = micros();
 Serial.println(gecenZaman);// Program başladıktan sonra geçen zamanı (us) ekrana yaz.
  delay(1000); // 1 saniye bekle. (Saniyede bir ekrana yazdır.)
}
```

2.7.5. Gelişmiş Giriş Çıkış İşlemleri

Arduino'nun pinlerden belirli bir frekansta kare dalga üretmek, dalga süresini hesaplamak veya kaydırmalı giriş çıkış (shift register) için kullanılan fonksiyonlar vardır.

tone() Fonksiyonu

tone() fonksiyonu Arduino dijital pini üzerinden belirtilen frekansta (%50 duty cycle) kare dalga üretmek için kullanılır. Süre belirtilebilir. Süre belirtilmezse noTone() fonksiyonuna ulaşılıncaya kadar kare dalga sinyal üretmeye devam eder. Pin çıkışına pizeo buzzer (pasif buzzer) veya hoparlör bağlanabilir. Üç adet parametre alır. Geriye değer döndürmez. Aynı anda birden fazla pinden çıkış veremez. tone() fonksiyonu (Mega dışındaki kartlar için) 3 ve 11 No.lu pinlerdeki PWM çıkışını engeller. tone() fonksiyonuyla en az 31 Hz değerinde pulse çıkış elde edilebilir.

```
tone(pin, frekans)

tone(pin, frekans, sure)

pin: Pin numarası.

frekans: Frekans.

sure: Frekansı üretme süresi.
```

Örnek: Ambulans sesi çıkarır.

```
byte buzzer = 7;
int sure = 500;
void setup() {
}
void loop() {
  tone(buzzer, 700);
  delay(sure);
  tone(buzzer, 900);
  delay(sure);
}
```

Örnek: Siren sesi çıkarır.

```
byte buzzer = 7; // Buzzer 7 numaralı pine bağlı.
void setup() {
}
void loop() {
  for (int frekans = 500; frekans < 2000; frekans += 10) { // Frekansı artır.
    tone(buzzer, frekans);
    delay(10);
  }
  for (int frekans = 2000; frekans > 500; frekans -= 10) { // Frekansı azalt.
    tone(buzzer, frekans);
    delay(10);
  }
}
```


noTone() Fonksiyonu

tone() fonksiyonuyla tetiklenen kare dalga sinyali durdurmak için kullanılır. Sinyali durdurulmak istenen pin numarası parametre olarak alınır. Geriye herhangi bir değer döndürmez. Birçok pinde tone() kullanılcaksa noTone() kullanıldıktan sonra diğerine geçilir.

```
noTone(pin)
```

pin: Pin numarası.

Örnek: Bazı notaları sırayla çalar.

```
void setup() {
}
void loop() {
  noTone(8);
  tone(6, 440, 200);
  delay(200);
  noTone(6);
  tone(7, 494, 500);
  delay(500);
```

```
  noTone(7);
  tone(8, 523, 300);
  delay(300);
}
```

pulseIn() Fonksiyonu

Giriş olarak ayarlanan pin üzerine uygulanan sinyal darbesinin kaç mikrosaniye süreyle HIGH ya da LOW durumunda kaldığını belirlemek için kullanılır. Geriye unsigned long türünde mikrosaniye cinsinden değer döndürür. Örnek olarak HIGH parametresi almışsa pin üzerine HIGH geldiğinde zamanlayıcıyı çalıştırır. Sinyal LOW olunca zamanlayıcıyı durdurur. HIGH seviyesinde kalma süresini mikrosaniye cinsinden unsigned long veri tipinde döndürür. Zaman aşımıyla belirtilen süre içerisinde pals sinyali konum değiştirmemişse geriye değer olarak 0 döndürür (Görsel 2.30).

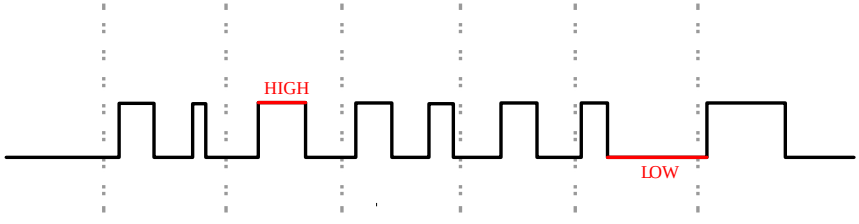
```
pulseIn(pin, durum)

pulseIn(pin, durum, zamanAsimi)

pin: Pin numarası.

durum: Okunacak pals tipi (HIGH veya LOW)

zamanAsimi: Darbe sinyalinin başlamasını bekleme süresi. Varsayılan süre 1 saniye
```



Görsel 2.30: HIGH veya LOW seviyelerinde kalma süresi

Örnek: Program butona basılma süresini vermektedir.

```
int pin = 7; // Butona bağlı.
unsigned long sure;
void setup() {
  Serial.begin(9600);
  pinMode(pin, INPUT_PULLUP); // Butonun diğer ucu GND'ye bağlı.
}
void loop() {
  sure = pulseIn(pin, LOW, 60000000); // Zaman aşımı 1 dakika ayarlandı.
  Serial.print(sure / 1000); // Butona basılma süresini milisaniye olarak göster.
Serial.println(" ms.");
}
```

Örnek: pulseIn() fonksiyonuyla yapılmış debounce (buton arkını önleme) programıdır.

```
int pin = const byte buton = 2;
unsigned long sure;
void setup() {
  Serial.begin(9600);
  pinMode(buton, INPUT_PULLUP);
}
void loop() {
  sure = pulseIn(buton, LOW) / 1000;
  if ( sure > 60)
    Serial.println("Butona basıldı.");
}
```

Örnek: Tek butonla LED'i yakıp söndürür.

```
const byte buton = 2;
const byte LED = 13;
unsigned long sure;
bool tersle = 1; // ilk basışta LED'i yakmak için.
void setup() {
  pinMode(buton, INPUT_PULLUP);
  pinMode(LED, OUTPUT);
}
void loop() {
  sure = pulseIn(buton, LOW) / 1000;
  if ( sure > 60) { // İlk 60 millisaniye içindeki bounce (ark) etkisinden kurtul.
    digitalWrite(LED, tersle); // Çıkışın konumunu değiştir.
    tersle = !tersle;
  }
}
```

pulseInLong() Fonksiyonu

Kullanımı pulseIn() fonksiyonuyla aynı olan pulseInLong(), uzun darbe sürelerini algılama ve kesmelerde (intterrupt) etkilenen senaryoları işlemede daha iyi olan bir alternatiftir.

Örnek: Ultrasonik sensörle mesafe ölçer.

```
const byte trig = 12; // Sinyal çıkışı.
const byte echo = 11; // Sinyal girişi.
unsigned long sure;
int uzaklik;
void setup() {
 pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  Serial.begin(9600);
}
void loop() {
  digitalWrite(trig, 0); // Kare dalga gönder.
```

```
  delayMicroseconds(10);
  digitalWrite(trig, 1);
  delayMicroseconds(10);
  sure = pulseInLong(echo, 1); // Gelen sinyalin HIGH'da kalma süresini oku.
  uzaklik = (0.0343 * sure) / 2;
  Serial.print(uzaklik);
  Serial.println(" cm");
}
```

shiftIn() Fonksiyonu

Bir byte veride her seferinde bir bit kaydırır. En yüksek değerlikli veya en düşük değerlikli bitten başlar. Her bit için clock pini HIGH olur. Bir sonraki bit veri hattından okunur ve ardından clock pini LOW olur. Dijital giriş sayısını arttırmak için kullanılır.

```
shiftIn(dataPin, clockPin, bitSirasi
```

dataPin: Her bir biti okuyan giriş pini.

clockPin: Clock sinyalini gönderen pin.

bitSirasi: Bit okuma sırası. MSBFIRST veya LSBFIRST.

shiftOut() Fonksiyonu

Her saat darbesinde bir byte verinin bir **bit**ini kaydırarak çıkışa gönderir. Dijital çıkış sayısını arttırmak için kullanılır.

```
shiftOut(dataPin, clockPin, bitSirasi, veri)
```

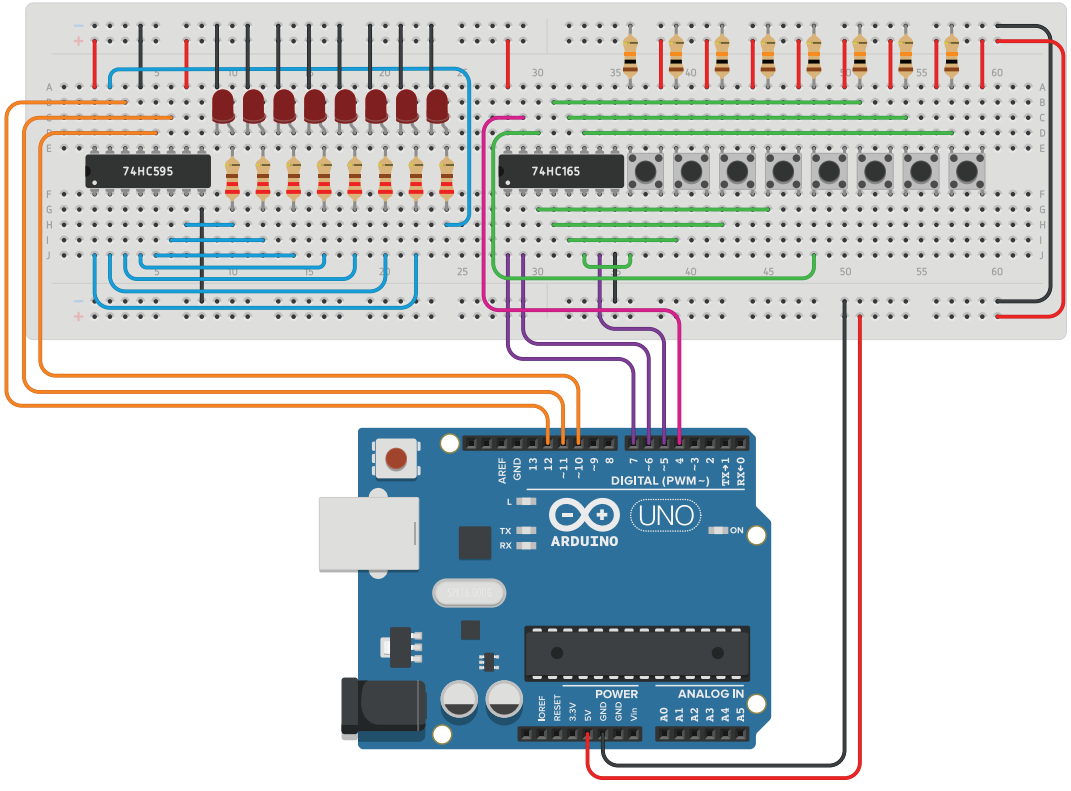
dataPin: Her bir biti gönderen çıkış pini.

clockPin: Clock sinyalini gönderen pin.

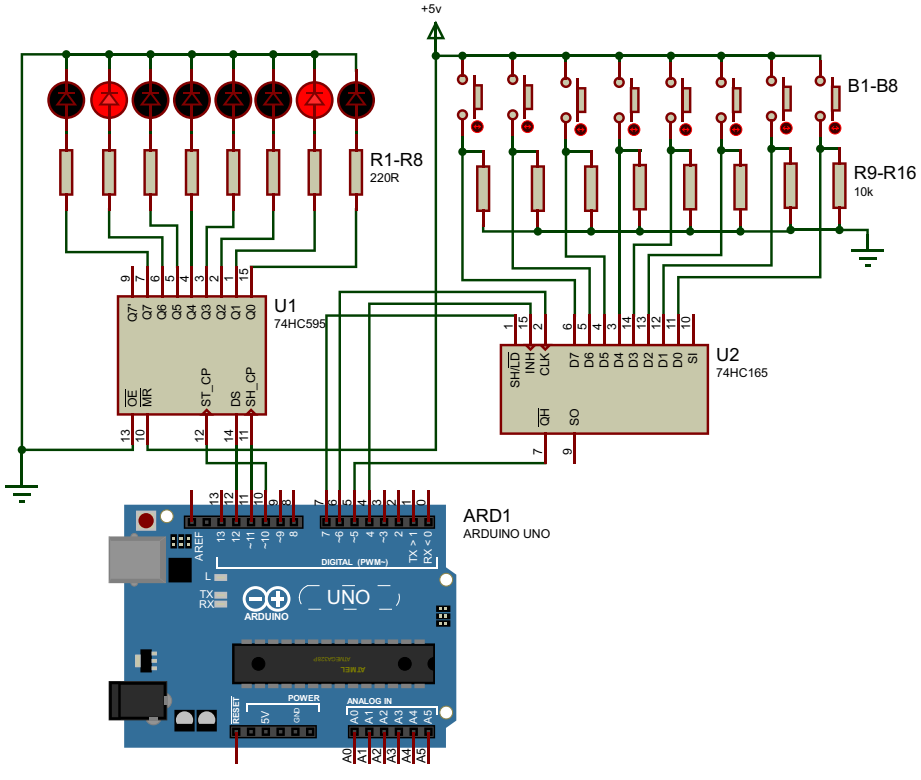
bitSirasi: Bit okuma sırası. MSBFIRST veya LSBFIRST.

veri: Kaydırılarak çıkışa aktarılacak bir byte veri.

Örnek: 74HC165 ve 74HC595 ile dijital giriş ve çıkış sayısını arttırmayı gösterir. Dijital giriş ve çıkış sayısını arttırma devresi görsel 2.31'de, şeması ise görsel 2.32'de görülmektedir.



Görsel 2.31: 74HC165 ve 74HC595 ile dijital giriş ve çıkış sayısını artırma devresi



Görsel 2.32: 74HC165 ve 74HC595 ile dijital giriş ve çıkış sayısını artırma şeması

```
int yukleme = 7;   // 74HC165 bağlantıları.
int clockAktif = 4;
int veriGiris = 5;
int clockGiris = 6;

const int bekle = 10; // 74HC595 bağlantıları.
const int clockPin = 11;
const int veri = 12;
int ledler[8];
void setup () {
  pinMode(yukleme, OUTPUT); // 74HC165 pinleri.
  pinMode(clockAktif, OUTPUT);
  pinMode(clockGiris, OUTPUT);
  pinMode(veriGiris, INPUT);

  pinMode(bekle, OUTPUT); // 74HC595 pinleri.
  pinMode(clockPin, OUTPUT);
  pinMode(veri, OUTPUT);
}
void loop() {
  digitalWrite(yukleme, LOW); // Girişleri oku.
  delayMicroseconds(5);
  digitalWrite(yukleme, HIGH);
  delayMicroseconds(5);
 digitalWrite(clockGiris, HIGH); // 74HC165'ten bilgiyi al.
  digitalWrite(clockAktif, LOW);
  byte gelenVeri = shiftIn(veriGiris, clockGiris, LSBFIRST);
  digitalWrite(clockAktif, HIGH);

  switch (gelenVeri) {
    case B11111110:
      ledler[0] = B11111111;
      ledler[1] = B01111110;
      ledler[2] = B10111101;
      ledler[3] = B11011011;
      ledler[4] = B11100111;
      ledler[5] = B11011011;
      ledler[6] = B10111101;
      ledler[7] = B01111110;
      break;
    case B11111101:
      ledler[0] = B00000001;
       ledler[1] = B00000010;
      ledler[2] = B00000100;
      ledler[3] = B00001000;
```

```
    ledler[4] = B00010000;
    ledler[5] = B00100000;
    ledler[6] = B01000000;
    ledler[7] = B10000000;
    break;
  case B11111011:
    ledler[0] = B10000001;
    ledler[1] = B01000010;
    ledler[2] = B00100100;
    ledler[3] = B00011000;
    ledler[4] = B00000000;
    ledler[5] = B00100100;
    ledler[6] = B01000010;
    ledler[7] = B10000001;
    break;
  case B11110111:
    ledler[0] = B10101010;
    ledler[1] = B01010101;
    ledler[2] = B10101010;
    ledler[3] = B01010101;
    ledler[4] = B10101010;
    ledler[5] = B01010101;
    ledler[6] = B10101010;
    ledler[7] = B01010101;
    break;
  case B11101111:
    ledler[0] = B10000000;
    ledler[1] = B00000001;
    ledler[2] = B01000000;
    ledler[3] = B00000010;
    ledler[4] = B00100000;
    ledler[5] = B00000100;
    ledler[6] = B00010000;
    ledler[7] = B00001000;
    break;
  case B11011111:
    ledler[0] = B11000000;
    ledler[1] = B01100000;
    ledler[2] = B00110000;
    ledler[3] = B00011000;
   ledler[4] = B00001100;
    ledler[5] = B00000110;
    ledler[6] = B00000011;
    ledler[7] = B10000001;
    break;
```

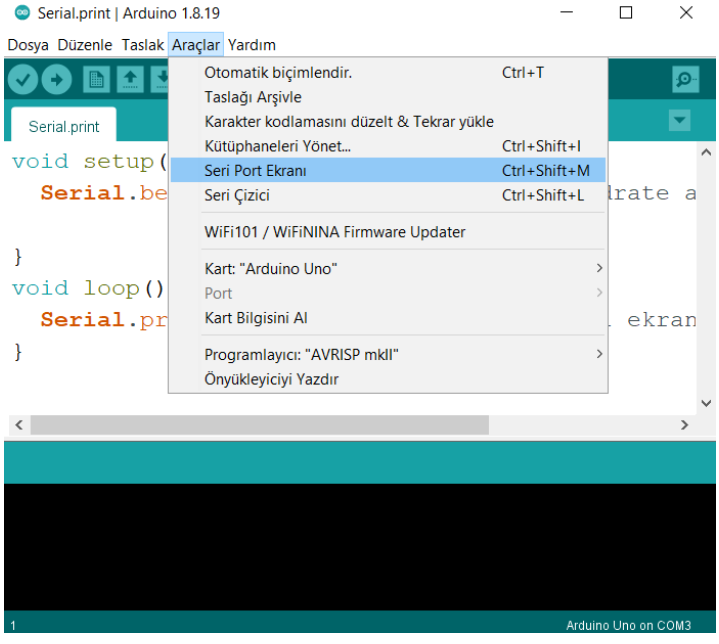
```
      case B10111111:
        ledler[0] = B11100000;
        ledler[1] = B01110000;
        ledler[2] = B00111000;
        ledler[3] = B00011100;
        ledler[4] = B00001110;
        ledler[5] = B00000111;
        ledler[6] = B10000011;
        ledler[7] = B11000001;
        break;
      case B01111111:
        ledler[0] = B10001000;
        ledler[1] = B01000100;
        ledler[2] = B00100010;
        ledler[3] = B00010001;
        ledler[4] = B10001000;
        ledler[5] = B01000100;
        ledler[6] = B00100010;
        ledler[7] = B00010001;
        break;
      default:
        break;
    }
    for (int i = 0; i < 8; i++) { // LED'lere gönder.
      digitalWrite(bekle, LOW);
      shiftOut(veri, clockPin, MSBFIRST, ledler[i]);
      digitalWrite(bekle, HIGH);
      delay(200);
    }
}
```

2.8. SERİ PORT İŞLEMLERİ

Arduino kartıyla bilgisayar veya diğer cihazlar arasındaki iletişim için kullanılır. Tüm Arduino kartlarında en az bir seri bağlantı noktası (UART veya USART olarak da bilinir.) bulunur.

Arduino Uno, Nano, Mini ve Mega'da bilgisayarla iletişim için 0 (Rx) ve 1 (Tx) pinleri kullanılır. Bu pinlere herhangi bir şey bağlanırsa program yükleme aşamasında hata verir.

Arduino'nun USB kablosu üzerinden (Rx ve Tx pinleri) bilgisayara veri aktarmak için Serial() fonksiyonu kullanılır. Görsel 2.33'te görüldüğü gibi Araçlar→Seri Port Ekranı seçeneğiyle açılır.



Görsel 2.33: Seri port ekranını açma

Serial.begin() Fonksiyonu

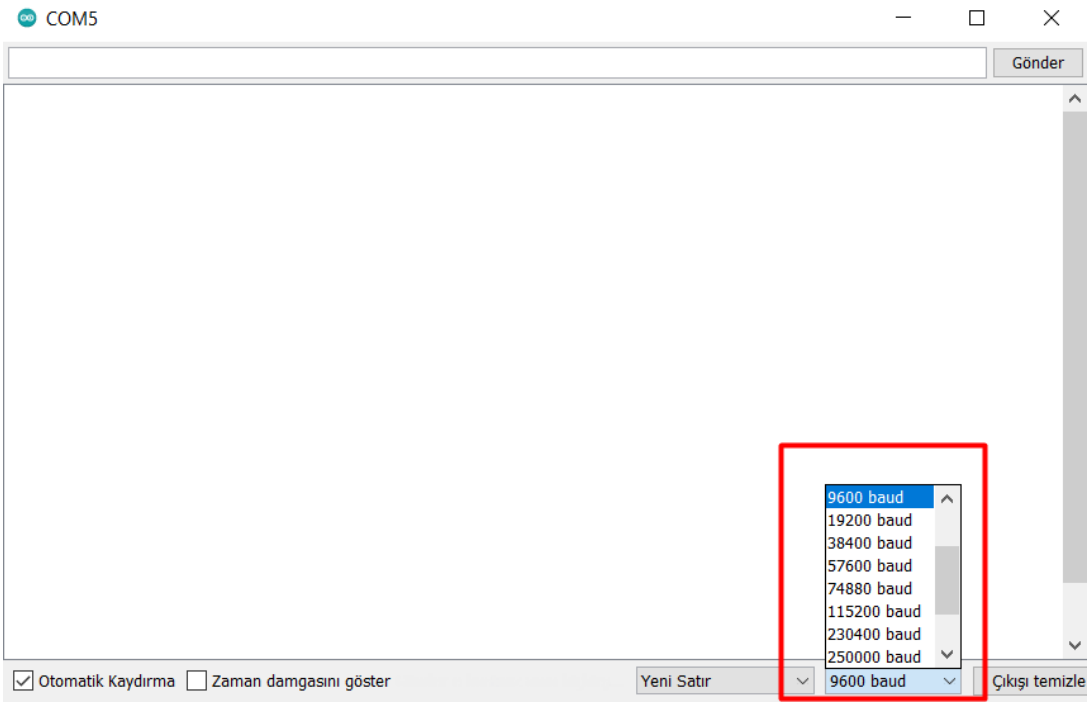
Serial.begin(9600) fonksiyonuyla seri port haberleşme hızı ayarlanır. 9600 bps (bit per second-saniyedeki bit sayısı) buadrate olarak isimlendirilen veri transfer hızıdır. Serial.begin() fonksiyonu içine yazılan değerle Görsel 2.34'te gösterilen seri port ekranındaki değer aynı şekilde ayarlanmalıdır.

NOT Programda Serial.begin() fonksiyonu kullanılıyorsa devrede 0. ve 1. pinlerde bağlantı olmamasına dikkat edilmelidir.

```
Serial.begin(hiz)
Serial.begin(hiz, ayar)
```

Hız: Seri portun saniyede göndereceği ve alacağı bit sayısıdır. Seri portun haberleşeceği cihazla aynı olmak zorundadır.

Ayar: Veri, eşlik ve durdurma bitlerini ayarlar.



Görsel 2.34: Seri port ekranı

Serial.print() Fonksiyonu

Seri port ekranına veri yazdırmak için kullanılır.

`Serial.print(deger)`

deger: herhangi bir veri tipindeki değer.

Örnek: Bazı veri tiplerini ekrana yazdırır.

```
Serial.print(35); // “35”
Serial.print(1.23456); // “1.23”
Serial.print(‘N’); // “N”
Serial.print(“Arduino.”); // “Arduino.”
```

Serial.print(deger, format)

format: Yazdırma formatı.

Örnek: Sayıları formatlı yazdırır.

```
Serial.print(78, BIN); // “1001110”
Serial.print(78, OCT); // “116”
Serial.print(78, DEC); // “78”
Serial.print(78, HEX); // “4E”
Serial.print(1.23456, 0); // “1”
Serial.print(1.23456, 2); // “1.23”
Serial.print(1.23456, 4); // “1.2345”
```

Serial.println() Fonksiyonu

Serial.print() fonksiyonuyla aynı özelliklere sahiptir. Tek farkı satır başı yapmasıdır.

F()makro Kullanımı: Serial.print("Arduino."); kullanıldığında yazdırılacak metin (dizi) normal olarak RAM'e kaydedilir. Progam (taslak) seri ekrana çok fazla şey yazdırıyorsa RAM hızlıca dolar. Boş flash bellek (programın yüklendiği hafıza) alanı varsa Serial.print(F("Arduino.")); söz dizimi kullanılarak **metin flash**a kaydedilir.

NOT DEFTERİ

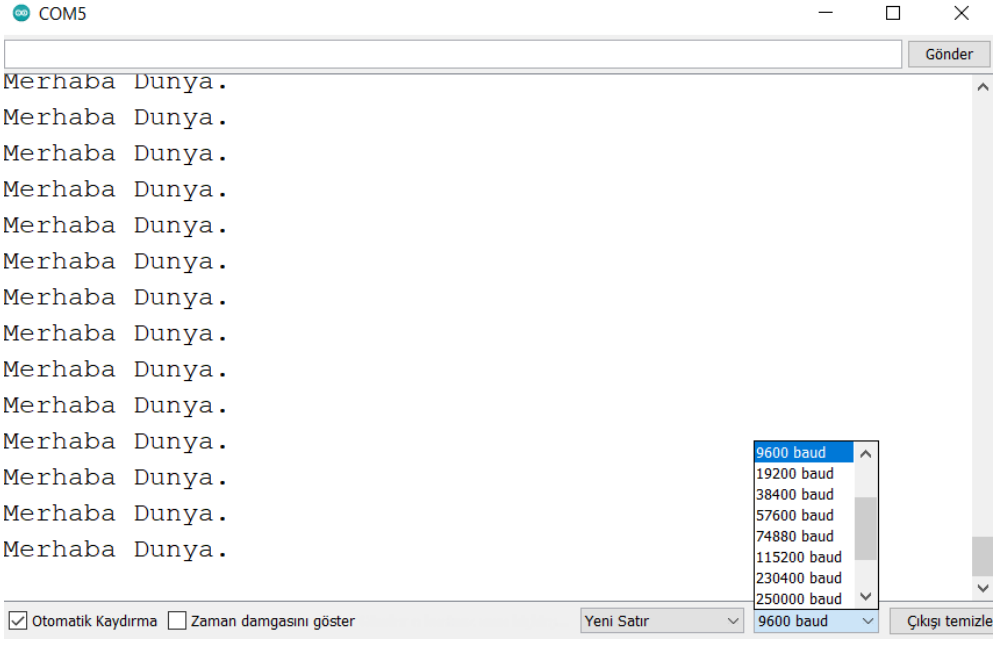


Uygulama Adı: Seri Port Ekran Uygulaması — No.: 9

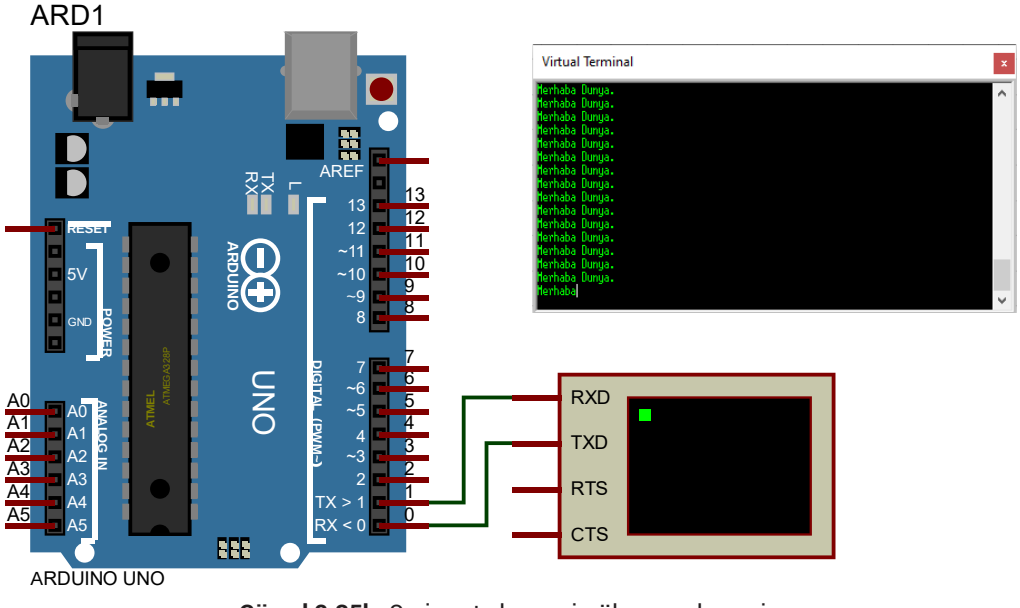
Amaç: **Seri port ekran uygulaması yapmak.**

Görsel 2.35a ve 2.35b'deki çıktılarda seri port ekranına Serial.println() fonksiyonuyla "Merhaba Dünya" yazdırılmaktadır.

İkinci programda butonla dijital giriş uygulamasında butonun durumu seri port ekranına yazdırılmaktadır (Görsel 2.36).



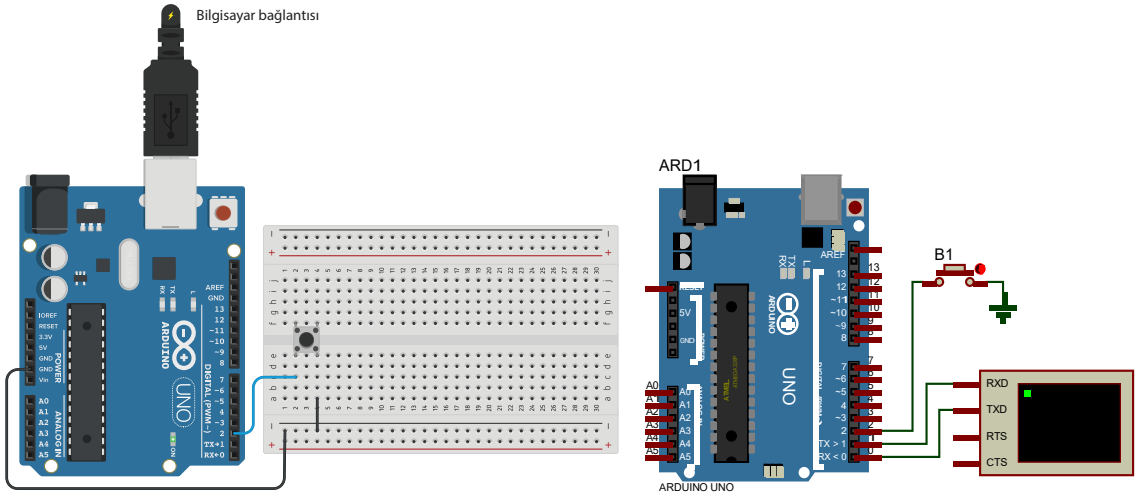
Görsel 2.35a: Seri port ekranı



Görsel 2.35b: Seri port ekranı simülasyon devresi

Seri port ekran uygulama programı aşağıdaki gibidir:

```
void setup() {Serial.begin(9600); // Seri iletişim baudrate ayarı yapıldı.
}
void loop() {
  Serial.println("Merhaba Dünya."); // Seri ekrana metin yazdırıldı.
}
```



Görsel 2.36: Buton durumunu seri port ekranına yazdırma devresi

Buton durumunu seri port ekranına yazdırma uygulaması programı aşağıdaki gibidir:

```
const  byte buton = 2;
void setup() {
  Serial.begin(9600); // Seri iletişim 9600 bps olarak ayarlandı.
  pinMode(buton, INPUT_PULLUP);
}
void loop() {
  boolean butonDurum = digitalRead(buton);
  Serial.println(butonDurum); // butonDurum değişkeni içindeki bilgi seri ekrana yazdırıldı.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. "Merhaba Dünya." programını yazıp Arduino'ya yükleyiniz. Seri port ekranına yazdırınız.
4. Seri port ekranındaki **baudrate'i** programdakinden farklı seçip sonuçları gözlemleyiniz.
5. Buton durumu programını yazıp Arduino'ya yükleyiniz. Buton durumunu seri port ekranına yazdırınız.
6. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Butonun durumunu **if** kullanarak seri port ekranına "Butona basıldı.", "Butona basılmadı." şeklinde yazdırınız.

Sorular

1. Programdaki **buadrate** hızıyla seri port ekranında seçili olan **buadrate** hızı farklı olursa ne olur? Açıklayınız.

2. Digital giriş çıkış uygulamalarında 0 ve 1 numaralı pinler neden tercih edilmemektedir? Açıklayınız.

3. Serial.print() fonksiyonuyla Serial.println() fonksiyonları arasındaki fark nedir? Açıklayınız.

4. Aşağıdaki çıktıyı veren programı yazınız.

E, dec: 69, hex: 45, oct: 105, bin: 1000101
F, dec: 70, hex: 46, oct: 106, bin: 1000110
G, dec: 71, hex: 47, oct: 107, bin: 1000111
H, dec: 72, hex: 48, oct: 110, bin: 1001000
I, dec: 73, hex: 49, oct: 111, bin: 1001001
J, dec: 74, hex: 4A, oct: 112, bin: 1001010
K, dec: 75, hex: 4B, oct: 113, bin: 1001011
L, dec: 76, hex: 4C, oct: 114, bin: 1001100
M, dec: 77, hex: 4D, oct: 115, bin: 1001101
N, dec: 78, hex: 4E, oct: 116, bin: 1001110
O, dec: 79, hex: 4F, oct: 117, bin: 1001111
P, dec: 80, hex: 50, oct: 120, bin: 1010000
Q, dec: 81, hex: 51, oct: 121, bin: 1010001
R, dec: 82, hex: 52, oct: 122, bin: 1010010
S, dec: 83, hex: 53, oct: 123, bin: 1010011
T, dec: 84, hex: 54, oct: 124, bin: 1010100
U, dec: 85, hex: 55, oct: 125, bin: 1010101
V, dec: 86, hex: 56, oct: 126, bin: 1010110
W, dec: 87, hex: 57, oct: 127, bin: 1010111
X, dec: 88, hex: 58, oct: 130, bin: 1011000
Y, dec: 89, hex: 59, oct: 131, bin: 1011001
Z, dec: 90, hex: 5A, oct: 132, bin: 1011010

Serial.available() Fonksiyonu

Seri iletişim arabelleğine (64 bayt) ulaşmış ve depolanmış olan verilerin byte (karakter) sayısını verir. Parametre almaz. Gelen karakter sayısını döndürür.

Örnek: Gelen verinin byte sayısını yazdırır.

```
void setup() {
  Serial.begin(9600); // Seri iletişim başlatıldı.
}
void loop() {
  if (Serial.available() > 0) { // Eğer buffer'da veri varsa...
    byte uzunluk = Serial.available(); // Gelen byte sayısını uzunluk değişkenine ata.
    Serial.println(uzunluk); // Uzunluk değişkeninin içeriğini yazdır.
    delay(1000);
  }
}
```

Serial.read() Fonksiyonu

Parametre almaz. Seri porttan gelen karakteri okur. Gelen karakter bir değişkene atanabilir.

Örnek: Seri porttan karakter okur.

```
void setup() {
  Serial.begin(9600); // Seri iletişim başlatıldı.
}
void loop() {
  if (Serial.available()>0) { // Seri iletişimde veri mevcutsa 1 yoksa 0.
  char karakter = Serial.read();// Karakter değişkenine seri hattan gelen veriyi yükle.
    Serial.print("Gelen karakter "); // Ekrana "Gelen karakter " yaz.
    Serial.println(karakter); // Karakter değişkeninin içeriğini yazdır.
  }
}
```

Serial.readString() Fonksiyonu

Seri porttan gelen verileri string (metin) olarak okur. Parametre almaz. Değişkene atanabilir.

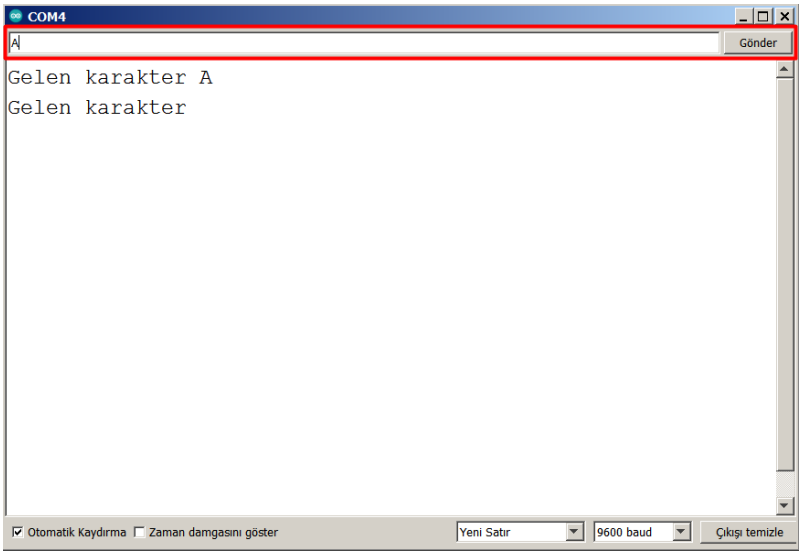
Örnek: Seri porttan karakter dizisi okuyarak seri port ekranına yazar.

```
String metin;
void setup() {
  Serial.begin(9600); // Seri iletişimi başlat
}
void loop() {
  if (Serial.available()) { // Veri geliyorsa...
    metin = Serial.readString(); // Gelen veriyi metin değişkenine ata.
    Serial.print("Gelen metin--->  ");
    Serial.println(metin);
  }
}
```

Uygulama Adı: Bilgisayardan Arduino'ya Veri Gönderme No.: 10

Amaç: **Bilgisayardan Arduino'ya veri gönderme uygulaması yapmak.**

Bu uygulamada Serial.read() fonksiyonuyla USB kablo üzerinden bilgisayardan gelen veriler okunarak seri port ekranına yazdırılmıştır. Bilgisayardan Arduino'ya veri göndermek için seri port ekranının üst kısmındaki kutuya karakter yazıp gönder düğmesine tıklanır (Görsel 2.37). Çok karakterli metin bilgisini okumak için Serial.readString() fonksiyonu kullanılır.



Görsel 2.37: Bilgisayardan Arduino'ya veri gönderme

Bilgisayardan Arduino'ya veri (tek karakter) gönderme programı aşağıdaki gibidir:

```
oid setup() {
  Serial.begin(9600); // Seri iletişim başlatıldı.
}
void loop() {
  if (Serial.available()>0) { // Seri iletişimde veri mevcutsa 1 yoksa 0.
    char karakter = Serial.read(); // Karekter değişkenine seri hattan gelen veriyi yükle.
    Serial.print("Gelen karakter "); // Ekrana "Gelen karakter " yaz.
    Serial.println(karakter); // Karakter değişkeninin içeriğini yazdır.
  }
}
```

Bilgisayardan Arduino'ya veri (metin) gönderme programı aşağıdaki gibidir:

```
String metin;
void setup() {
  Serial.begin(9600); // Seri iletişimi başlat.
}
void loop() {
  if (Serial.available()) { // Veri geliyorsa...
```

```
    metin = Serial.readString(); // Gelen veriyi metin değişkenine ata.
    Serial.print("Gelen metin--->  ");
     Serial.println(metin);
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Bilgisayardan Arduino'ya veri (tek karakter) gönderme programını yazınız.
4. Seri port ekranına A karakterini yazıp enter tuşuna basınız.
5. Bilgisayardan Arduino'ya veri (metin) gönderme programını yazınız.
4. Seri port ekranına adınızı yazıp enter tuşuna basınız.

SIRA SİZDE

1. LED'le dijital çıkış uygulamasındaki LED'i bilgisayardan A karakteri göndererek yakıp B karakteri göndererek söndürünüz.

```
char karakter;
void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600);// Seri iletişimi başlat.
}
void loop() {
  if (Serial.available()) { // Veri geliyorsa...
    karakter = Serial.read(); // Gelen veriyi karakter değişkenine ata.
    if (karakter == 'A') // Gelen karakter A'sa...
      digitalWrite(13, 1); // LED'i yak.
    if (karakter == 'B') // Gelen karakter B'se...
      digitalWrite(13, 0); // LED'i söndür.
  }
}
```

2. String (metin); değişkeni tanımlayarak, Serial.readString() fonksiyonuyla "yak" ve "söndür" metinlerini arka arkaya göndererek LED'i yakıp söndürünüz.

İPUCU Yukarıdaki programı kullanarak Arduino'ya USB kablo yerine bluetooth modül bağlayıp, kablosuz iletişimle LED'i yakıp söndürebilirsiniz.

Sorular

1. Karakter kabul eden programa metin bilgisi gönderildiğinde nasıl işlem yapar? Açıklayınız.
2. Bilgisayarda A ile a farklı karakterler midir? Belirtiniz.
3. "Arduino öğreniyorum." metninde kaç karakter vardır? Boşluk karakteri klavyedeki hangi tuşla yazılır? Belirtiniz.

Serial.end() Fonksiyonu

Seri portu devre dışı bırakarak RX ve TX pinlerinin dijital giriş çıkış olarak kullanılmasına izin verir. Seri portu tekrar etkinleştirmek için Serial.begin() fonksiyonu kullanılır. Serial.end() fonksiyonu parametre almaz ve değer döndürmez.

```
Serial.end()
```

Serial.readBytes() Fonksiyonu

Seri porttan gelen verilerin hepsini veya bir kısmını char veya **byte** dizisine atar.

```
Serial.readBytes(veriDizisi, uzunluk)
```

veriDizisi: Arabellekteki (buffer) karakterlerin (**byte**) atandığı dizi. Veri tipi char veya **byte** dizisi.

uzunluk: Dizinin uzunluğu. Veri tipi int.

Örnek: Program seri porttan Arduino'ya gönderilen verinin ikinci karakterini ekrana yazdırır.

```
char karakter;
void setup() {
  Serial.begin(9600); // Seri iletişim başlatıldı.
}
void loop() {
  if (Serial.available() > 0) { // Eğer buffer'da veri varsa...
    int uzunluk = Serial.available(); // Gelen byte sayısını uzunluk değişkenine ata.
   char gelenVeri[uzunluk]; // Gelen veri uzunluğunda "gelenVeri" adında dizi oluştur.
  Serial.readBytes(gelenVeri, uzunluk);// Verinin tamamını, "gelenVeri" dizisine aktar.
karakter = gelenVeri[1];// Dizinin ikinci karakterini (ASCII) karakter değişkenine ata.
  }
Serial.println(karakter);// Karakter değişkenin içindeki char tipindeki karakteri yaz.
  delay(1000);
}
```

Örnek: Verinin ilk üç **byte**ı okunur.

```
void loop() {
  if (Serial.available() > 0) { // Eğer buffer'da veri varsa...
   char gelenVeri[2]; // 3 byte uzunluğunda "gelenVeri" adında dizi oluştur.
Serial.readBytes(gelenVeri, 2);// Verinin ilk üç karakterini, "gelenVeri" dizisine al.
}
```

Serial.readBytesUntil() Fonksiyonu

Seri porttan gelen verileri belli bir karaktere kadar okur. Belirlenen karaktere ulaşıldığında okuma işlemi sona erer ve belirtilen karakter arabellekten temizlenir.

```
Serial.readBytesUntil(karakter, veriDizisi, uzunluk)
```

karakter: Gelen veride aranan karakter. Veri tipi char.

veriDizisi: Arabellekteki karakterlerin atandığı dizi. Veri tipi char veya **byte** dizisi.

uzunluk: Dizinin uzunluğu. Veri tipi int.

Örnek: Program seri porttan Arduino'ya gönderilen verinin A karakterine kadar olan karakter sayısını ekrana yazdırır.

```
int kacKarakter;
void setup() {
  Serial.begin(9600);
}
void loop() {
  if (Serial.available() > 0) { // Eğer buffer'da veri varsa...
    int uzunluk = Serial.available(); // Gelen byte sayısını uzunluk değişkenine ata.
    char gelenVeri[uzunluk]; // Gelen veri uzunluğunda "gelenVeri"  dizisi oluştur.
    kacKarakter = Serial.readBytesUntil('A', gelenVeri, uzunluk);
    // Gelen veride A karakterine kadar kaç karakter var?
    Serial.println(kacKarakter); // A karakterine kadar olan karakter sayısını yazdır.
    delay(1000);
  }
}
```

Serial.readStringUntil() Fonksiyonu

Seri porttan gelen metni belirlenen karaktere kadar okur. Belirlenen karaktere kadar olan metni döndürür.

```
Serial.readStringUntil(karakter)
```

karakter: Gelen metinde aranan karakter. Veri tipi char.

Örnek: A karakterine kadar olan kısmı alır.

```
String metin;
void setup() {
  Serial.begin(9600); // Seri iletişimi başlat.
}
void loop() {
  if (Serial.available()) { // Veri geliyorsa...
    metin = Serial.readStringUntil('A'); // A karakterine kadar olan kısmı metin de-
ğişkenine ata.
    Serial.print("Gelen metin--->  ");
    Serial.println(metin);
  }
}
```

Serial.write() Fonksiyonu

Seri porta binary veri gönderir. Gönderilen veri **byte** veya **byte** dizisi şeklindedir.

```
Serial.write(deger)
Serial.write(metin)
Serial.write(veriDizisi, uzunluk)
```

deger: Gönderilen bir byte veri.

metin: Gönderilen birçok byte veri.

veriDizisi: Dizi şeklinde gönderilen veri.

uzunluk: Dizinin byte uzunluğu.

Örnek: Seri porttan bir **byte** veri gönderir.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  Serial.write(45); // 45 değerindeki bir byte veri gönder.
  int uzunluk = Serial.write("Arduino");
  // Arduino metnini gönder ve metnin byte uzunluğunu değişkene ata.
}
```

Serial.availableForWrite() Fonksiyonu

Serial.write() işlemini engellemeden seri arabelleğe yazmak için kullanılabilecek **byte** (karakter) sayısını verir.

```
Serial.availableForWrite()
```

Örnek: Seri port arabelleğinde kalan boşluk (**byte** sayısı) seri port ekranına yazdırır.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  Serial.write("Arduino");
  int kalanByteUzunlugu = Serial.availableForWrite();
  Serial.println(kalanByteUzunlugu);
  // Seri porta "Arduino" verisini gönderdikten sonra 64 byte hafızadan kalanı yazdır.
  delay(1000);
}
```

Serial.find() Fonksiyonu

Seri porttan gelen veride belirlenen karakteri arar. Char veri tipinde parametre alır. Fonksiyon karakteri bulursa true, bulamazsa false sonucunu döndürür.

```
Serial.availableForWrite()
Serial.find(karakter)
Serial.find(karakter, uzunluk)
```

karakter: Aranan karakter. Veri tipi char.

uzunluk: Belirtilen uzunlukta (sayıda) karakter içinde arama yapar.

```
void setup() {
  Serial.begin(9600);
```

```
}
void loop() {
  if (Serial.available() > 0) {
    bool ara = Serial.find('a');// Gelen veri içinde a karakterini ara.
    Serial.println(ara);// Gelen veride a karakteri varsa 1 yoksa 0 yazar.
  }
}
```

Serial.findUntil() Fonksiyonu

Seri porttan gelen veride sonlandırıcı karaktere kadar belirlenen karakteri arar. Fonksiyon karakteri bulursa true, bulamazsa false sonucunu döndürür.

```
Serial.findUntil(karakter, sonlandirici)
```

karakter: Aranan karakter. Veri tipi char.

sonlandırıcı: Aramanın sonlanacağı karakter. Veri tipi char.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  if (Serial.available() > 0) {
    bool ara = Serial.findUntil("a",".");// '.' (nokta) karakterine kadar 'a' karakterini ara.
    Serial.println(ara);// 'a' karakteri varsa 1 yoksa 0 yazar.
  }
}
```

Serial.flush() Fonksiyonu

Seri porttan giden verinin iletiminin tamamlanmasını beklemek için kullanılır. Parametre almaz ve geriye değer döndürmez.

Serial.parseInt() Fonksiyonu

Seri porttan gelen veriler arasında tamsayı olanları alır. Rakam dışındaki ASCII karakterler atlanır. Long veri tipinde tamsayı döndürür. Rakam bulunmazsa 0 döndürür.

```
Serial.parseInt()
```

Örnek: Seri porttan gelen veriler arasında tam sayı olanları alır.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  if (Serial.available() > 0) {
    int sayi = Serial.parseInt();// Gelen veride tamsayı varsa sayi değişkenine ata.
    Serial.print("Gelen veri içindeki sayı: ");
    Serial.println(sayi);
  }
}
```

Örnek: Bir kısmı verilen Arduino bluetooth araç programında, akıllı telefon uygulamasından bluetooth ile gönderilen karakterlerden a,b,c ve d yönü belirlerken tam sayı gönderildiğinde **hiz** değişkenine atanmaktadır.

```
void loop() {
  if (bluetooth.available()) {
     karakter = bluetooth.read();// Seri iletişimden gelen karakteri oku.
     hiz = bluetooth.parseInt();// Gelen veri tamsayıysa hiz değişkenine ata.
    if (karakter == 'a') { // Gelen karakter a'sa…
      // ileri
      analogWrite(IN1, hiz);
      analogWrite(IN2, 0);
      analogWrite(IN3, hiz);
      analogWrite(IN4, 0);
    } else if (karakter == 'b') {
      // geri
          // …
```

Serial.parseFloat() Fonksiyonu

Seri porttan gelen veriler arasında ondalıklı sayıları alır. Float veri tipinde ondalıklı sayı döndürür. Ondalıklı sayı bulunmazsa 0 döndürür.

```
Serial.parseInt()
```

Örnek: Ondalıklı sayıyı bulur.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  if (Serial.available() > 0) {
    float sayi = Serial.parseFloat();// Ondalıklı sayıyı bul sayi değişkenine ata.
    Serial.print("Gelen veri içindeki sayı: ");
    Serial.println(sayi); // sayi değişkeni float veri tipindedir.
  }
}
```

Serial.setTimeout() Fonksiyonu

Seri porttan gelecek olan verinin en fazla ne kadar süre bekleneceği milisaniye cinsinden ayarlanır. Varsayılan ayar 1000 milisaniyedir.

```
Serial.setTimeout(zaman)
```

zaman: Milisaniye cinsinden beklenecek süre.

Serial.setTimeout() fonksiyonuyla ayarlanan süreyi aşağıdaki fonksiyonlar kullanır.

```
Serial.find()
Serial.findUntil()
Serial.parseInt()
```

```
Serial.parseFloat()
Serial.readBytes()
Serial.readBytesUntil()
Serial.readString()
Serial.readStringUntil()
```

Örnek: Program metin girişi için 10 saniye bekler.

```
void setup() {
  Serial.begin(9600);
  Serial.println("Lütfen adınızı giriniz");
  Serial.setTimeout(10000);
  String metin = Serial.readString(); // 10 saniye içinde gelen metni oku.
  if (metin.length() > 0) { // Metin değişkeninde veri varsa…
    Serial.print("Gelen metin--->  ");
    Serial.println(metin);
  } else  Serial.print("Belirlenen süre içinde giriş yapmadınız.");
}
void loop() {
  // Giriş bilgisi setup fonksiyonunda bir kez istendi.
}
```

Serial.peek() Fonksiyonu

Serial.read() foksiyonuyla benzerdir. Gelen verinin sadece ilk **byte**ını okur. Okunan veri arabellekten silinmez. Parametre almaz. Gelen verinin ilk **byte**ını döndürür.

```
Serial.peek()
void setup() {
  Serial.begin(9600);
}
void loop() {
  if (Serial.available() > 0) {
    char karakter = Serial.peek();// Karakter değişkenine seri hattan gelen veriyi yükle.
    Serial.print("Gelen karakter "); // Ekrana "Gelen karakter " yaz.
    Serial.println(karakter); // Karakter değişkeninin içeriğini yazdır.
  }
}
```

SerialEvent() Fonksiyonu

Ana programdan bağımsız olarak (kesme gibi) çalışır. Seri porttan veri geldiğinde aktif olur ve fonksiyon içine yazılan komutları çalıştırır. Parametre almaz. Değer döndürmez.

```
void serialEvent() {
  // Seri porttan veri geldiği anda buradaki komutlar çalışır.
}
```

Örnek: Program blink uygulamasını seri porttan A karakteri geldiğinde çalıştırır, B karakteri geldiğinde durdurur.

```
char karakter;
bool kontrolBiti;
void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  if (karakter == 'A')
    kontrolBiti = 1;
  if (karakter == 'B')
    kontrolBiti = 0;
  if (kontrolBiti == 1)
  {
    digitalWrite(13, HIGH);
    delay(1000);
    digitalWrite(13, LOW);
    delay(1000);
  }
}
void serialEvent() { // Seri porttan gelen veri varsa aşağıdaki kodları çalıştır.
  karakter = Serial.read(); // Gelen karakteri oku.
}
```

Karakter Fonksiyonları

Harf, rakam, noktalama işaretleri ve sembollerden oluşan karakterler üzerinde işlem yapan fonksiyonlardır. Bu fonksiyonların tümü parametre olarak tek karakter alır ve geriye true veya false döndürür.

isAlpha(): Karakter harf ise true döndürür.

```
isAlpha(karakter)
```

karakter: Char veri tipinde değişken.

Örnek: Karakter bulur.

```
char karakter1 = 'A';
char karakter2 = 9;
void setup() {
  Serial.begin(9600);
  if (isAlpha(karakter1)) {
    Serial.println("karakter1 harftir");
  } else {
    Serial.println("karakter1 harf değildir");
  }
  if (isAlpha(karakter2)) {
    Serial.println("karakter2 harftir");
```

```
  } else {
    Serial.println(“karakter2 harf değildir”);
  }
}
void loop() {}
```

isAlphaNumeric(): Karakter alfanümerikse (harf veya rakam) true döndürür.

Örnek: Alfanümerik karakter bulur.

```
char karakter1 = ‘A’;
char karakter2 = 9;
char karakter3 = ‘9’;
void setup() {
  Serial.begin(9600);
  if (isAlphaNumeric(karakter1)) {
    Serial.println(“karakter1 alfanümerik karakterdir.”);
  } else {
    Serial.println(“karakter1 alfanümerik karakter değildir.”);
  }
  if (isAlphaNumeric(karakter2)) {
    Serial.println(“karakter2 alfanümerik karakterdir.”);
  } else {
    Serial.println(“karakter2 alfanümerik karakter değildir.”);
  }
  if (isAlphaNumeric(karakter3)) {
    Serial.println(“karakter3 alfanümerik karakterdir.”);
  } else {
    Serial.println(“karakter3 alfanümerik karakter değildir.”);
  }
}
void loop() {}
```

isAscii(): Karakter ASCII'se [American Standard Code for Information Interchange (ımerıkın standırd kod for informeyşın intırçeync)] true döndürür. ASCII karakterler 7 bitlik bir karakter kümesidir. 33 tane basılmayan (ekranda çıkmayan) kontrol karakteri ve 95 tane basılan karakter bulunur.

`isAscii(karakter)`

karakter: Char veri tipinde değişken.

Örnek: ASCII karakter bulur.

```
char karakter1 = ‘A’;
char karakter2 = 9;
char karakter3 = ‘9’;
char karakter4 = ‘#’;
char karakter5 = ‘☺’; // Klavyeden Alt+3 giriniz. (3’ü Num Lock’tan giriniz.)
```

```
void setup() {
  Serial.begin(9600);
  if (isAscii(karakter1)) {
    Serial.println("karakter1 ASCII karakterdir.");
  } else {
    Serial.println("karakter1 ASCII karakter değildir.");
  }
  if (isAscii(karakter2)) {
    Serial.println("karakter2 ASCII karakterdir.");
  } else {
    Serial.println("karakter2 ASCII karakter değildir.");
  }
  if (isAscii(karakter3)) {
    Serial.println("karakter3 ASCII karakterdir.");
  } else {
    Serial.println("karakter3 ASCII karakter değildir.");
  }
  if (isAscii(karakter4)) {
    Serial.println("karakter4 ASCII karakterdir.");
  } else {
    Serial.println("karakter4 ASCII karakter değildir.");
  }
  if (isAscii(karakter5)) {
    Serial.println("karakter5 ASCII karakterdir.");
  } else {
    Serial.println("karakter5 ASCII karakter değildir.");
  }
}
void loop() {}
```

isControl(): Karakter kontrol karakteriyse (ASCII'nin ilk 33 karakteri) true döndürür.

`isControl(karakter)`

karakter: Char veri tipinde değişken.

Örnek: Kontrol karakteri bulur.

```
char karakter1 = 'A';
char karakter2 = 9;
void setup() {
  Serial.begin(9600);
  if (isControl(karakter1)) {
    Serial.println("karakter1 kontrol karakteridir.");
  } else {
    Serial.println("karakter1 kontrol karakteri değildir.");
```

```
  }
  if (isControl(karakter2)) {
    Serial.println("karakter2 kontrol karakteridir.");
  } else {
    Serial.println("karakter2 kontrol karakteri değildir.");
  }
}
void loop() {}
```

isDigit(): Karakter rakamsa true döndürür.

`isDigit(karakter)`

karakter: Char veri tipinde değişken.

Örnek: Rakam bulur.

```
char karakter1 = '9';
char karakter2 = 9;
void setup() {
  Serial.begin(9600);
  if (isDigit(karakter1)) {
    Serial.println("karakter1 rakam karakteridir.");
  } else {
    Serial.println("karakter1 rakam karakteri değildir.");
  }
  if (isDigit(karakter2)) {
    Serial.println("karakter2 rakam karakteridir.");
  } else {
    Serial.println("karakter2 rakam karakteri değildir.");
  }
}
void loop() {}
```

isGraph(): Karakter yazdırılabilir karakterse true döndürür.

`isGraph(karakter)`

karakter: Char veri tipinde değişken.

Örnek: Yazdırılabilir karakter bulur.

```
char karakter1 = 'A';
char karakter2 = 9;
void setup() {
  Serial.begin(9600);
  if (isGraph(karakter1)) {
    Serial.println("karakter1 yazdırılabilir karakterdir.");
  } else {
```

```
    Serial.println("karakter1 yazdırılabilir karakter değildir.");
  }
  if (isGraph(karakter2)) {
    Serial.println("karakter2 yazdırılabilir karakterdir.");
  } else {
    Serial.println("karakter2 yazdırılabilir karakter değildir.");
  }
}
void loop() {}
```

isHexadecimalDigit(): Karakter heksadesimal rakamsa (A-F, 0-9) true döndürür.

isHexadecimalDigit(karakter)

karakter: Char veri tipinde değişken.

Örnek: Heksadesimal rakam bulur.

```
char karakter1 = '11A';
char karakter2 = '5G';
void setup() {
  Serial.begin(9600);
  if (isHexadecimalDigit(karakter1)) {
    Serial.println("karakter1 heksadesimal karakterdir.");
  } else {
    Serial.println("karakter1 heksadesimal karakter değildir.");
  }
  if (isHexadecimalDigit(karakter2)) {
    Serial.println("karakter2 heksadesimal karakterdir.");
  } else {
    Serial.println("karakter2 heksadesimal karakter değildir.");
  }
}
void loop() {}
```

isLowerCase(): Karakter küçük harfse true döndürür.

isLowerCase(karakter)

karakter: Char veri tipinde değişken.

isUpperCase(): Karakter büyük harfse true döndürür.

isUpperCase(karakter)

karakter: Char veri tipinde değişken.

Örnek: Küçük veya büyük karakter bulur.

```
char karakter1 = 'a';
char karakter2 = 'A';
void setup() {
  Serial.begin(9600);
```

```
  if (isLowerCase(karakter1)) {
    Serial.println("karakter1 küçük harftir.");
  } else {
    Serial.println("karakter1 küçük harf değildir.");
  }
  if (isUpperCase(karakter2)) {
    Serial.println("karakter2 büyük harftir.");
  } else {
    Serial.println("karakter2 büyük harf değildir.");
  }
}
void loop() {}
```

isPrintable(): Karakter yazıcıda basılabilirse true döndürür.

`isPrintable(karakter)`

karakter: Char veri tipinde değişken.

Örnek: Yazıcıda basılabilir karakter bulur.

```
char karakter1 = 65; // 'A' ile aynıdır.
char karakter2 = 29;
void setup() {
  Serial.begin(9600);
  if (isPrintable(karakter1)) {
    Serial.println("karakter1 basılabilir.");
  } else {
    Serial.println("karakter1 basılamaz.");
  }
  if (isPrintable(karakter2)) {
    Serial.println("karakter2 basılabilir.");
  } else {
    Serial.println("karakter2 basılamaz.");
  }
}
void loop() {}
```

isPunct(): Karakter noktalama işaretiyse true döndürür.

`isPunct(karakter)`

karakter: Char veri tipinde değişken.

Örnek: Noktalama işareti bulur.

```
char karakter1 = '?';
char karakter2 = 'A';
void setup() {
  Serial.begin(9600);
```

```
  if (isPunct(karakter1)) {
    Serial.println("karakter1 noktalama işaretidir.");
  } else {
    Serial.println("karakter1 noktalama işareti değildir.");
  }
  if (isPunct(karakter2)) {
    Serial.println("karakter2 noktalama işaretidir.");
  } else {
    Serial.println("karakter2 noktalama işareti değildir.");
  }
}
void loop() {}
```

isSpace(): Karakter boşluk karakteriyse true döndürür. Boşluk karakterleri space, form beslemesi (‹\f›), yeni satır (‹\n›), satır başı (‹\r›), yatay sekme (‹\t›) veya dikey sekme (‹\v›) karakterleridir.

`isSpace(karakter)`

karakter: Char veri tipinde değişken.

Örnek: Boşluk karakterleri bulur.

```
char karakter1 = ' ';
char karakter2 = 'A';
void setup() {
  Serial.begin(9600);
  if (isSpace(karakter1)) {
    Serial.println("karakter1 boşluk karakteridir.");
  } else {
    Serial.println("karakter1 boşluk karakteri değildir.");
  }
  if (isSpace(karakter2)) {
    Serial.println("karakter2 boşluk karakteridir.");
  } else {
    Serial.println("karakter2 boşluk karakteri değildir.");
  }
}
void loop() {}
```

isWhitespace(): Karakter boşluk karakteri (space decimal 32) veya tab karakteriyse (\t) true döndürür.

`isWhitespace(karakter)`

karakter: Char veri tipinde değişken.

Örnek: Boşluk karakterleri bulur.

```
char karakter1 = ' ';
char karakter2 = 'A';
```

```
void setup() {
  Serial.begin(9600);
  if (isSpace(karakter1)) {
```

Bit ve Byte Fonksiyonları

Bit ve **byte** fonksiyonları bit seviyesinde işlem yapmak için kullanılan fonksiyonlardır.
bit(): Girilen bitin onluk değerini döndürür.

```
Sebit(n)
n: Hesaplanacak bit.
```

Örnek: On basamaklı binary sayının onluk basamaktaki sınırını verir.

```
void setup() {
Serial.begin(9600);
Serial.println(bit(10)); // 1024 döndürür.
}
void loop() {}
```

bitClear(): Sayının bir **bit**ini siler (0 yapar.). Yeni sayıyı döndürür.

isbitClear(x, n)

x: Biti silinecek sayı.

n: Silinecek **bit** numarası. En düşük anlamlı (sağ) **bit**ten başlar.

Örnek: x ve n int tipinde iki sayıdır. 6'nın ikilik karşılığı 0110'dır. n=1 olduğundan sağdan ikinci **bit** 0 olur. İkilik sayı 0100 olur. Onluk karşılığı 4'tür.

```
void setup() {
  Serial.begin(9600);
  int x = 6;
  int n = 1;
  Serial.print(bitClear(x, n)); // Sonuç 4.
}

void loop() {}
```

bitSet(): Sayının bir **bit**ini 1 yapar. Yeni sayıyı döndürür.

bitSet(x, n)

x: Bitlerinden biri 1 yapılacak sayı.

n: 1 yapılacak **bit** numarası. En düşük anlamlı (sağ) **bit**ten başlar.

Örnek: Sekiz LED'li kara şimşek programıdır.

```
byte i;
int sure = 100;
byte sayi = 255; // İkilik B11111111›dir.
void setup() {
  DDRD = sayi; // 0-7 numaralı pinler çıkış olarak ayarlandı.
}
```

```
void loop() {
 for (i = 0; i <= 7; i++)  {
    PORTD = bitSet(sayi, i); // Sayının bitlerini sırayla 1 yap.
    delay(sure);
    PORTD = bitClear(sayi, i); // Sayının bitlerini sırayla 0 yap.
  }
  for (i = 6; i > 0; i--) { // İlk ve son LED dönüşlerde bir kez yanar.
    PORTD = bitSet(sayi, i);
    delay(sure);
    PORTD = bitClear(sayi, i);
  }
}
```

bitRead(): Sayının bir **bit**ini okur. **Bit**in değerini (0 veya 1) döndürür.

bitRead(x, n)

x: Biti okuncak sayı.

n: Okunacak **bit** numarası. En düşük anlamlı (sağ) **bit**ten başlar.

Örnek: Sekiz LED'li animasyon programıdır.

```
byte j, i;
int sure = 100;
byte sayilar[] = {129, 66, 36, 24, 24, 36, 66, 129};
void setup() {
  for (i = 0; i <= 7; i++)  {
    pinMode(i, OUTPUT); // 0 - 7 numaralı pinleri çıkış yap.
  }
}
void loop()
{
  for (j = 0; j <= 7; j++) {
    for (i = 0; i <= 7; i++) {
      digitalWrite(i, bitRead(sayilar[j], i));// Sıradaki sayının i'inci bitini çıkışa yaz.
    }
    delay(sure);
  }
}
```

bitWrite(): Sayının bir **bit**ini değiştirir. **Bit**in değerini (0 veya 1) döndürür.

bitWrite(x, n, b)

x: Biti değiştirilecek sayı.

n: Değiştirilecek **bit** numarası. En düşük anlamlı (sağ) **bit**ten başlar.

b: Bitin değeri (0 veya 1) .

```
void setup() {
  Serial.begin(9600);
  byte x = 0b10000000;
  Serial.println(x, BIN); // 10000000
```

```
  bitWrite(x, 0, 1);  // x sayısının 0. bitine 1 yaz.
  Serial.println(x, BIN); // 10000001
}
void loop() {}
```

highByte(): İki **byte**lık sayının soldaki yüksek değerlikli **byte**ını verir. İki **byte**tan büyük sayılarda sağdan (LSB) ikinci **byte**ı verir.

```
bithighByte(x)
```

x: Herhangi bir veri tipindeki sayı.

lowByte(): İki **byte**lık sayının sağdaki düşük değerlikli **byte**ını verir. İki **byte**tan büyük sayılarda en sağdaki (LSB) **byte**ı verir.

```
lowByte(x)
```

x: Herhangi bir veri tipindeki sayı.

2.9. ANALOG GİRİŞ ÇIKIŞ İŞLEMLERİ

Arduino analog girişler için 10 **bit**lik (0 -1023) A0-A5 pinlerini kullanırken analog çıkışlar için dijital pinlerden ~ işareti olanları kullanır.

2.9.1. Analog Giriş İşlemleri

Analog işlemlerde kullanılan komutlar okuma, yazma ve referans olmak üzere üç çeşittir.

AnalogRead() Fonksiyonu

A0-A5'le belirtilen analog pinlerin değerini okur. Arduino kartları 10 **bit** analogdan dijitale dönüştürücü (ADC) içerir. 0 V ile 5 V arasındaki giriş voltajlarını 0 ile 1023 arasındaki tam sayı değerlerine eşler. Her bir adım başına 5/1024=0,0049 V (4,9 mV) düşmektedir. Fonksiyon girişteki gerilime göre 0-1023 arasında int veri tipinde değer döndürür. analogReference(EXTERNAL) fonksiyonuyla AREF pininden verilen 5V'tan düşük voltaj tepe değerini oluşturmaktadır. Örneğin AREF 4 V ise 1023 4 V'a eşlenir.

Uno, Nano, Mini, Mega'da bir analog girişi okumak yaklaşık 100 mikrosaniye (0,0001 sn.) sürer, bu nedenle maksimum okuma hızı saniyede yaklaşık 10.000 defadır.

```
analogRead(pin)
```
pin: Analog giriş pininin adı. Uno'da A0 - A5, Mini ve Nano'da A0 - A7, Mega'da A0 - A15.

Örnek: Tüm analog çıkışlı sensörler için kullanılan temel program.

```
const byte cikisPini = 13;
void setup() {
  Serial.begin(9600); //Seri iletişim başlatıldı.
  pinMode(cikisPini, OUTPUT);}
void loop() {
  int analogGirisDegeri = analogRead(A0); //A0 girişinden okunan 0-1023 arası veriyi
analogGirisDegeri değişkenine yükle.
  Serial.println(analogGirisDegeri); //analogGirisDegeri değişkeninin içeriğini yazdır.
  if (analogGirisDegeri < 800) { // Karşılaştırma satırı.
    digitalWrite(cikisPini, HIGH); // Karşılaştırma geçerliyse yapılacaklar.}
  else { // Karşılaştırma geçerliyse değilse yapılacaklar.
    digitalWrite(cikisPini, LOW);}  }
```

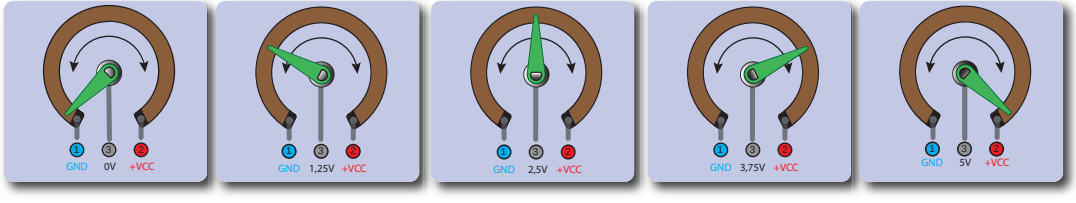
Uygulama Adı: Potansiyometreyle Analog Giriş Uygulaması No.: 11

Amaç: **Potansiyometreyle analog giriş uygulaması yapmak.**

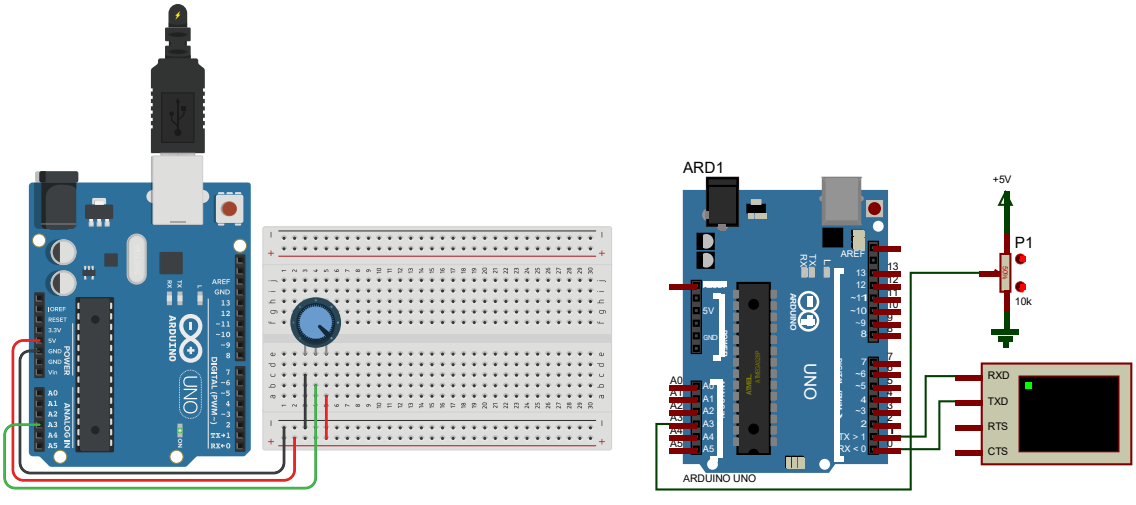
Görsel 2.38'de potansiyometrenin pin yapısı ve yapılan ayara göre orta ucundan alınan gerilim değerleri görülmektedir. Potansiyometrenin bir kenardaki ucu GND'ye, diğer kenardaki ucuysa 5 V'a bağlanmıştır. Gerilim bölücü olarak çalışan bu potansiyometrenin orta ucundan 0-5 V arasında bir gerilim elde edilir. Bu gerilim Arduino'nun A3 analog girişine uygulanmıştır (Görsel 2.39).

NOT

Analog girişleri pinMode() fonksiyonuyla giriş olarak tanımlamaya gerek yoktur.



Görsel 2.38: Potansiyometre pin yapısı



Görsel 2.39: Potansiyometreyle analog giriş şeması ve devresi

Potansiyometreyle analog giriş uygulama programı aşağıdaki gibidir:

```
void setup() {
  Serial.begin(9600); // Seri iletişim başlatıldı.
// Analog pinleri giriş olarak tanımlamaya gerek yoktur.
}
void loop() {

  int pot = analogRead(A3); // A3 girişinden okunan 10 bitlik veriyi pot değişkenine yükle.
```

Potansiyometreyle analog giriş uygulama programı aşağıdaki gibidir:

```
void setup() {
  Serial.begin(9600); // Seri iletişim başlatıldı.
// Analog pinleri giriş olarak tanımlamaya gerek yoktur.
}
void loop() {

  int pot = analogRead(A3); // A3 girişinden okunan 10 bitlik veriyi pot değişkenine yükle.
  Serial.println(pot); // Pot değişkeninin içeriğini yazdır.
}
```

Voltmetre uygulama programı aşağıdaki gibidir:

```
void setup() {
  Serial.begin(9600);
}
void loop() {

  int pot = analogRead(A3); /* A3 pininden 0 - 1023 arasında okunan sayıyı pot değişkenine yükle. */

   float gerilim = (5.0/1024.0) * pot; /* pot değerini ~0,0049 değeriyle çarp gerilim değişkenine yükle. 0.0 - 5.0 arasında küsuratlı sayılar elde edeceğimiz için gerilim değişkeni float olarak tanımlandı. */

 Serial.print(gerilim);
  Serial.println(" Volt");
}
```

DİKKAT İngilizcede küsuratlı sayılarda nokta (3.14) kullanılırken Türkçede virgül (3,14) kullanılmaktadır.

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Görsel 2.39'daki devreyi kurunuz.
3. Programı yazıp Arduino'ya yükleyiniz.
4. Potansiyometrenin GND'ye bağlı ucuyla orta ucunu, potansiyometrenin konumunu değiştirerek voltmetreyle ölçüm değerlerini gözlemleyiniz.

SIRA SİZDE

1. Blink uygulamasındaki LED'in yanma süresini potansiyometreyle değiştirecek şekilde programı yazınız.

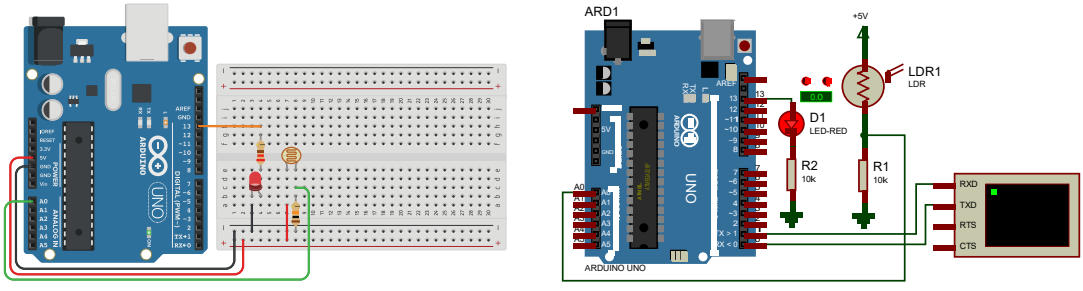
Sorular

1. Pot değişkeni neden **byte** yerine int tipinde tanımlanmıştır? Açıklayınız.
2. 30 V'luk gerilim kaynağı Arduino ile nasıl ölçülür? Belirtiniz.

Uygulama Adı: LDR'yle Analog Giriş Uygulaması — No.: 12

Amaç: **LDR'yle analog giriş uygulaması yapmak.**

LDR, ışıkla direnci ters orantılı olarak değişen elemandır. Arduino'nun 5 V gerilimi üzerine uygulanan LDR'nin direnç-gerilim değişiminin anlaşılabilmesi için 10 kΩ'luk bir direnç seri bağlanarak gerilim bölücü oluşturulur. Bağlantı noktasının gerilim değişimi Arduino'nun analog girişine verilir.



Görsel 2.40: LDR'yle analog giriş şeması ve devresi

LDR'yle analog giriş uygulama programı aşağıdaki gibidir:

```
const byte LED = 13; // LED değişkeninin içeriği 13'le sabitlendi.
int LDR;
void setup() {
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  LDR = analogRead(A0); // Analog giriş bilgisini LDR değişkenine yükle.

  if (LDR < 800)  // 800 değeri aydınlık/karanlık durumuna göre istenilen değerle değiştirilebilir.
    digitalWrite(LED, 1);
  else
    digitalWrite(LED, 0);
  Serial.println(LDR); // Seri port ekranından LDR'nin aldığı değer gözlenebilir.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.40'taki devreyi kurunuz.
4. Seri port ekranını açınız.
5. Elinizle LDR'nin üzerini kapatarak seri ekrandaki değeri ve devredeki LED'i gözlemleyiniz.
6. LED'i yakıp söndüren en uygun değeri programda güncelleyerek tekrar deneyiniz.
7. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Programı aydınlıkta LED yanacak şekilde düzenleyiniz.

2. Aşağıdaki program LDR'nin algıladığı ışık değişimini ses değişimi olarak vermektedir.

```
const byte buzzer = 13; // Buzzer 13 numaralı pine bağlı.
void setup() {
  pinMode(buzzer, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  int LDR = analogRead(A0); // Analog giriş bilgisini LDR değişkenine yükle.

  int frekans = LDR * 1; // LDR içindeki değeri 1'le çarp. (2, 3... ile çarpılarak frekans artırılabilir.)

  tone(buzzer, frekans); // 13 numaralı pinde frekans değişkeninin değerinde kare dalga (%50 duty cycle) oluşturur.

  Serial.println(frekans); // Seri port ekranından frekans gözlenebilir.
}
```

DİKKAT Analog giriş uygulamalarında analog çıkışlı sensör modülleri de kullanılabilir.

Sorular

1. 1023 değeri 5 V'sa 512 değeri kaç volttur? Hesaplayınız.

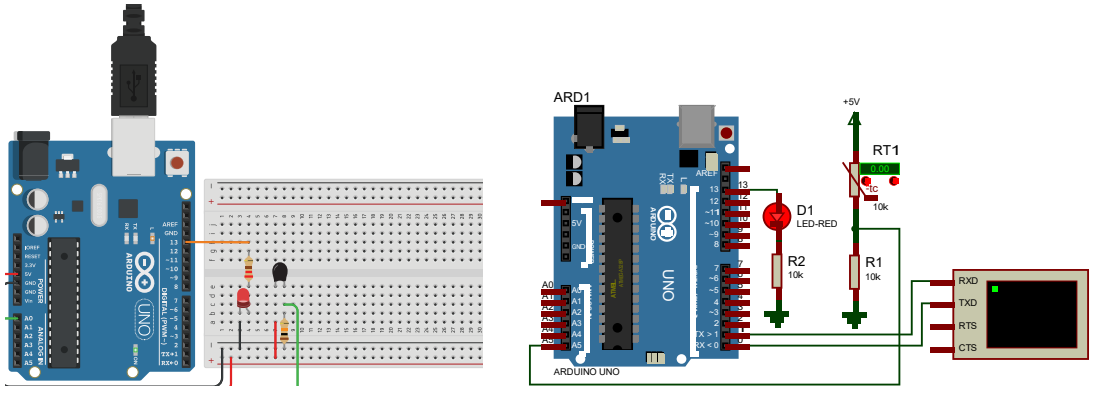
2. LDR değişkeninin değeri neden belli bir değerin altına inememektedir? Belirtiniz.

3. 10 kΩ'luk direnç 4,7 kΩ ile değiştirilirse gerilim bölücünün orta ucunda oluşan referans gerilimi 0 V ile 5 V'tan hangisine yaklaşır? Açıklayınız.

Uygulama Adı: NTC'yle Analog Giriş Uygulaması — No.: 13

Amaç: NTC'yle analog giriş uygulaması yapmak.

Görsel 2.41'deki devrede NTC ve NTC'ye seri bağlı direnç gerilim bölücü olarak çalışmaktadır. Oda sıcaklığında (yaklaşık 25 °C) NTC'nin direnci yaklaşık 8 kΩ-10 kΩ'dur. NTC'nin sıcaklığı artırıldığında direnci ve üzerine düşen gerilim azalır. Böylelikle 10 kΩ'luk sabit direnç üzerindeki gerilim artar. Direnç üzerindeki bu gerilim analog giriş tarafından algılanarak değeri 1023'e doğru yaklaşır. Bu değer belirlenen referans değerini aştığında uyarı LED'i yanar.



Görsel 2.41: NTC'yle analog giriş şeması ve devresi

NTC'yle analog giriş uygulama programı aşağıdaki gibidir:

```
const byte LED = 13, ntcPin = 19; // A5'e (19 numaralı pin) NTC bağlandı.
int ntcDegeri;
void setup() {
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  ntcDegeri = analogRead(ntcPin); // A5'ten gelen veriyi ntcDegeri değişkenine yükle.
  if (ntcDegeri > 800)  // 800'den büyükse...
    digitalWrite(LED, HIGH);
  else
    digitalWrite(LED, LOW);
  Serial.println(ntcDegeri); // Seri port ekranına ntcDegeri yazdır.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.41'deki devreyi kurunuz. Programı yükleyiniz.
4. NTC'yi ısıtınız.
5. Seri ekrandan analog değeri gözlemleyiniz.
6. Referans değerini farklı bir değerle değiştirerek uygulamayı tekrarlayınız.

SIRA SİZDE

1. PNTC'yle ortam sıcaklığını ölçmek mümkün müdür? Araştırınız.

```
#include <math.h> // Matematik kütüphanesini çağır.
const byte LED = 13;

void setup() {
  Serial.begin(9600); // Seri iletişimi başlat.
  pinMode(LED, OUTPUT); // 13 numaralı pini çıkış yap.
}
void loop() {

  int analogGirisDegeri = analogRead(A0); // A0'dan analog giriş değerini oku.

  // NTC için analog değer - sıcaklık dönüşüm formülü.
  double sicaklik = log(((10240000 / analogGirisDegeri) - 10000));
  sicaklik = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * sicaklik * sicaklik))
* sicaklik);
  sicaklik = sicaklik - 273.15;
  Serial.println(sicaklik); // Sıcaklığı ekrana yazdır.

  if (sicaklik > 30) { // Sıcaklık 30 dereceden yüksekse...
    digitalWrite(LED, HIGH);
  } else {
    digitalWrite(LED, LOW);
  }
  delay(500);
}
```

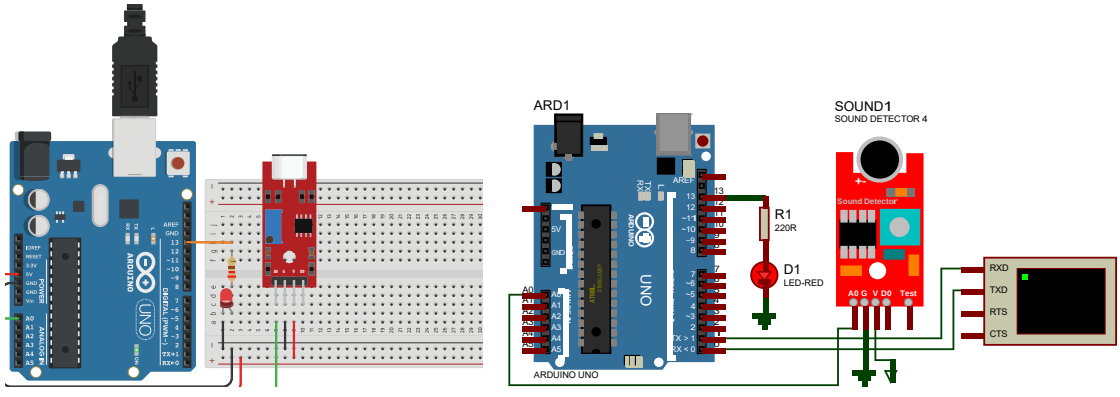
Soru

1. Analog giriş değeri 800'ü aştığında çıkıştaki motoru durdurma bilgisi göndermek için programda nasıl değişiklik yapılmalıdır? Belirtiniz.

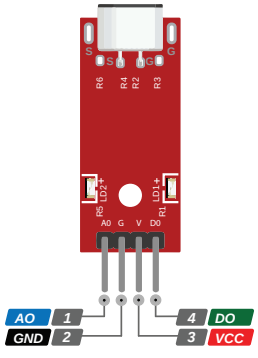
Uygulama Adı: Alkışla Çalışan LED Uygulaması No.: 14

Amaç: Alkışla çalışan LED uygulaması yapmak.

Görsel 2.42a'da alkışla çalışan LED uygulaması görülmektedir. Analog çıkışlı mikrofon modülünden gelen sinyal belli bir seviyeyi aştığında çıkıştaki LED yanmaktadır. Seviye tekrar aşıldığında LED sönmektedir. Görsel 2.42b'de ise mikrofon devresinin pin sıralaması verilmiştir.



Görsel 2.42a: Alkışla çalışan LED şeması ve devresi



Görsel 2.42b: Mikrofon modülü pin yapısı

A0: Analog veri çıkışı.
GND: Toprak pini.
VCC: 5 V besleme.
D0: Dijital veri çıkışı.

Alkışla çalışan LED uygulama programı aşağıdaki gibidir:

```
const byte led = 12;
int mic;
bool degistir = 1; // Boolean tipinde oluşturulan değiştir isimli değişkene 1 yüklendi.
void setup() {
  pinMode(led, OUTPUT);
  Serial.begin(9600);
}
void loop() {
```

```
  mic = analogRead(A0); // Analog girişten gelen veriyi (0 - 1023) mic değişkenine yükle.
  if (mic > 59) { // Seri ekrandan uygun referans noktası görülebilir.
    digitalWrite(led, degistir); // ilk seferde LED'i yakar.
    degistir = !degistir; // Boolean değişken olan degistir'i tersle.
    delay(100); // Debounce gecikmesi (Çok hızlı-kararsız yanıp sönmeyi engellemek için.)
  }
  Serial.println(mic); // Analog bilginin değerini göster.
}
    digitalWrite(LED, HIGH);
  else
    digitalWrite(LED, LOW);
  Serial.println(ntcDegeri); // Seri port ekranına ntcDegeri yazdır.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 2.42a'daki devreyi kurunuz. Programı yükleyiniz.
4. Seri ekrandan analog girişin aldığı değerleri gözlemleyiniz.
5. Alkış ve yüksek seslerde LED'in sırayla yanıp söndüğünü gözlemleyiniz.
6. Donanımınızı ses şiddetine göre referans değerini değiştirip tekrar deneyiniz.
7. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Alkışla yanan LED beş saniye sonra sönecek şekilde programı düzenleyiniz.

Sorular

1. `digitalWrite(13, !digitalRead(13));` komutu ne yapar? Araştırınız.
2. `digitalWrite(13, digitalRead(13)^1);` komutu ne yapar? Araştırınız.

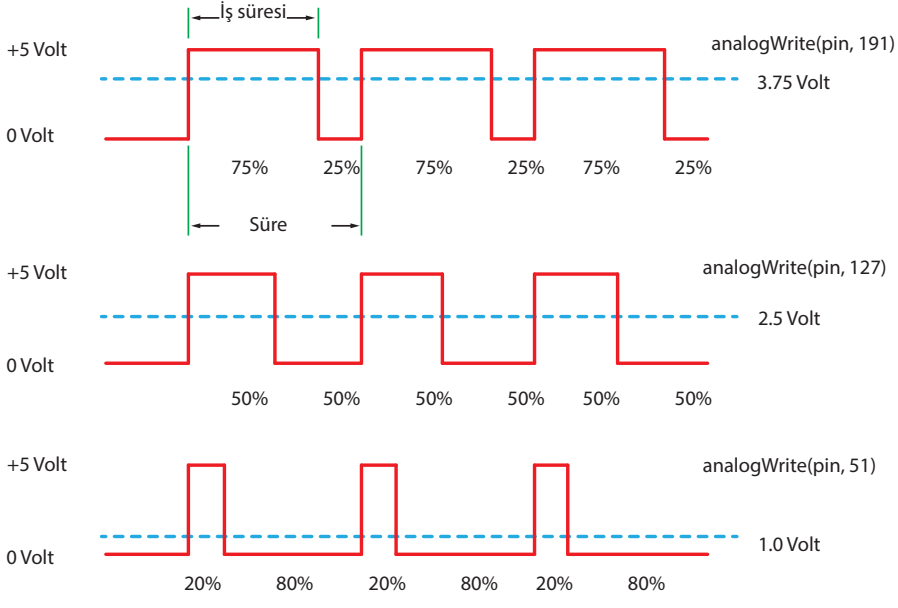
2.9.2. Analog Çıkış İşlemleri

Her mikrodenetleyicinin çıkışından gerçek anlamda analog gerilim alınamaz. Analog çıkış gerilimleri genellikle sayısal PWM sinyalleri ile elde edilir.

analogWrite() Fonksiyonu

Seçilen pinde PWM sinyalinin üretilmesini sağlar. PWM sinyalinin görev zamanı (duty cycle) fonksiyona verilen değerlerle belirlenir. Bir LED'i değişen parlaklıklarda yakmak veya bir motoru çeşitli hızlarda sürmek için kullanılabilir.

PWM [Pulse Width Modulation (puls wîdth mecıleyşın)] darbe genişliği modülasyonu, dijital bir sinyal olup (sürekli değişen 0-5 V), anahtarlamayla 5 V'ta kalma süresi (duty cycle) ayarlanarak kısmi analog sinyal oluşturmaktır. Arduino'da dijital analog dönüştürücü (DAC) 8 bit olduğundan çıkıştaki analog sinyal 256 (0-255) adımdan oluşur. Görsel 2.43'te analogWrite(pin, 51) verildiğinde duty cycle %20 ve çıkış 1 V'tur. analogWrite(pin, 127) verildiğinde duty cycle %50 ve çıkış 2,5 V'tur. analogWrite(pin, 191) verildiğinde duty cycle %75 ve çıkış 3,75 V'tur. Her bir adım 5/256=0,019 V'tur.



Görsel 2.43: PWM sinyali

analogWrite() fonksiyonu Uno , Nano, Mini'de 3, 5, 6, 9, 10 ve 11 numaralı pinlerde kullanılır. PWM frekansı 490 Hz'dir (5 ve 6. pinlerde 980 Hz.). Bu pinler pinMode() fonksiyonuyla çıkış olarak ayarlanmadan da analogWrite() fonksiyonuyla kullanılabilir.

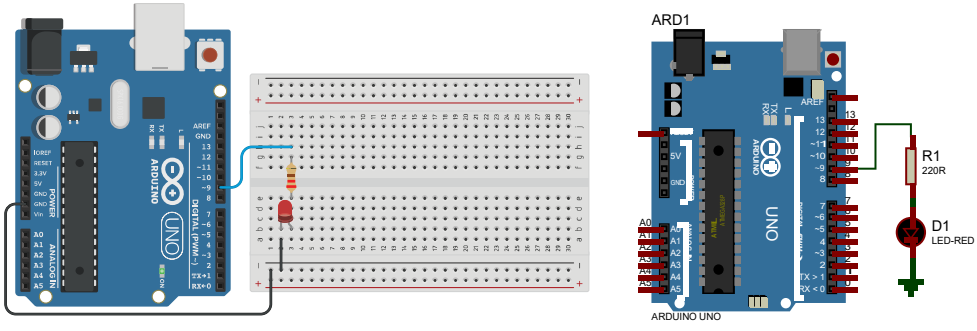
```
analogWrite(pin,deger)
```

pin: PWM özellikli pinlerden biri. Veri tipi int.
deger: 0-255 arası PWM değeri. Veri tipi int.

Uygulama Adı: Analog Çıkış Uygulaması — No.: 15

Amaç: Analog çıkış uygulaması yapmak.

Görsel 2.44'teki uygulamada LED 9 numaralı PWM özellikli pine bağlanmıştır. LED'in parlaklığını artırıp azaltmak için analogWrite() fonksiyonun ikinci parametresi 0-255 arası değiştirilir. 255'e yaklaştıkça parlaklık artmakta, 0'a yaklaştıkça parlaklık azalmaktadır.



Görsel 2.44: Analog çıkış şeması ve devresi

Analog çıkış uygulama programı aşağıdaki gibidir:

```
const byte ledPin = 9;     // LED 9 numaralı pine bağlı.
void setup() {
  // PWM pinleri çıkış olarak ayarlamaya gerek yoktur.
}
void loop() {
  for (int pwm = 0 ; pwm <= 255; pwm += 5) { // 0'dan 255'e 5'er artır.
    analogWrite(ledPin, pwm);
    delay(30);
  }
  for (int pwm = 255 ; pwm >= 0; pwm -= 5) { // 255'ten 0'a 5'er azalt.
    analogWrite(ledPin, pwm);
    delay(30);
  }
}
```

İşlem Basamakları

1. Görsel 2.44'teki devreyi kurunuz.
2. Programı yükleyerek LED'in parlaklığını gözlemleyiniz.
3. PWM'i birer adım arttırıp azaltarak deneyiniz.

SIRA SİZDE

1. Benzer bir programı for döngüsü kullanmadan yazınız.

```
const byte led = 9;
byte x, y = 0;
```

```
void setup() {
}
void loop() {
  if (y == 0) {
    x = x + 1;
    analogWrite(led, x);
    delay(10);
  }
  if (x == 255)
  {
    y = y - 1;
    analogWrite(led, y);
    delay(10);
  }
}
```

2. RGB LED'le 255^3 renk gösterimi (Ortak anot veya ortak katot kullanılabilir.).

```
const byte kirmizi=9, yesil=10, mavi=11; // PWM pinleri.
void setup() {
}
void loop() {
  for (int a = 0; a <= 255; a++) {
    analogWrite(kirmizi, a);
    for (int b = 0; b <= 255; b++) {
      analogWrite(yesil, b);
      for (int c = 0; c <= 255; c++) {
        analogWrite(mavi, c);
      }
    }
  }
// Kodu buraya yazınız.
}
```

3. **Önceki programda 255^3** renk gösterimi bittiğinde 1 saniye aralıklarla **kırmızı**, **yeşil** ve **mavi** yakacak kodu ekleyiniz.

Sorular

1. Hangi PWM değeri 1.25 V çıkış verir? Belirtiniz.

2. Analog sinyalde 0 V ile 5 V arasında sonsuz değer varken 8 **bit**lik çıkışta 0 V ile 5 V arasında kaç değer vardır? Açıklayınız.

Matematiksel Fonksiyonlar

abs(): Bir sayının mutlak değerini verir. Bir sayının mutlak değeri sıfıra olan uzaklığıdır. Parametre olarak sayı alır. Sayı sıfır veya pozitifse sayının kendisini, sıfırdan küçükse (negatifse) sayıyı pozitif olarak döndürür.

`abs(x)`

`x:` Sayı.

Örnek: Kod parçasında bluetooth üzerinden gelen negatif veri PWM çıkışlarına gönderilmektedir.

```
if(data[0] < -50)// geri
   {
     analogWrite(in1,0);
     analogWrite(in2,abs(data[0]));
     analogWrite(in3,0);
     analogWrite(in4,abs(data[0]));
   }
```

constrain(): Aralık sınırlamak için kullanılır. Üç adet parametre alır. Değişken sınırlar içindeyse kendisini, alt sınır altındaysa alt sınırı, üst sınır üstündeyse üst sınır değerini döndürür.

`bsconstrain(degisken, altSinir, ustSinir)`

`degisken:` Sınırlandırılacak değişken.

`altSinir:` Alt sınır değeri.

`ustSinir:` Üst sınır değeri.

Örnek: A0 girişinden gelen 0-1023 arasındaki değerlerin 50-150 arasındaki değerlerini kullanır.

```
void void setup() {
  Serial.begin(9600);
}
void loop() {
  int pot = analogRead(A0);
  int aralik = constrain(pot, 50, 150);
  Serial.println(aralik);
}
}
```

map(): Belli bir aralıktaki değeri başka bir aralığa dönüştürür. Beş adet parametre alır. Dönüştürülmüş aralıktaki değerleri int veri tipinde döndürür.

`map(deger, altDeger, ustDeger, yeniAltDeger, yeniUstDeger)`

`deger:` Dönüştürülecek değer.

`altDeger:` Dönüştürülmek istenen verinin alt değeri.

`ustDeger:` Dönüştürülmek istenen verinin üst değeri.

`yeniAltDeger:` Dönüştürülmüş alt değer.

Örnek: Aralığı tersten verir.

```
y = map(x, 1, 50, 50, 1);
```

Örnek: İvme sensöründen gelen -16384 ile 16384 arasındaki değerler -255 ile 255 aralığına dönüştürülmektedir.

```
data[0] = map(x, -17000, 17000, -255, 255 ); // X düzleminin verisi (ileri-geri)
data[1] = map(y, -17000, 17000, -255, 255); // Y düzleminin verisi (sağ-sol)
```

NOT

Eğer ondalık sayılarda dönüştürme yapılmak istenirse aşağıdaki map() fonksiyonunun matematiksel eş değeri olan işlem kodlarına dönüştürülmelidir.

$$y = \frac{(x - y_{min}) * (x_{max} - x_{min})}{(y_{max} - y_{min}) + x_{min}}$$

Örnek: 100'lük not sisteminde 70'in karşılığını 4'lük sistemde verir.

```
double donustur(double deger, double altDeger, double ustDeger, double yeniAltDeger, double yeniUstDeger)
{
  return (deger - altDeger) * (yeniUstDeger - yeniAltDeger) / (ustDeger - altDeger) + yeniAltDeger;
}
void setup() {
  Serial.begin(9600);
  Serial.println(map(70.0, 0.0, 100.0, 0.0, 4.0)); // Sonuç 2,80'dir.
  // Float ve double veri tiplerinde sayılar noktalı girilir.
}
void loop() {}
```

max(): İki sayıdan büyük olanını seçer (Döndürür.).

```
max(x, y)
```

x: Birinci sayı.

y: İkinci sayı.

Örnek: İki analog girişten büyük olanı seçer. 50'nin altındaki değerleri eler.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  int sensor1 = analogRead(A0);
  int sensor2 = analogRead(A1);
  int deger = max(sensor1, sensor2); // İki sensör değerinden büyük olanını seç.
  int sonuc = max(deger, 50); // Sonuç 50 veya üstü olur.
  Serial.println(sonuc);
}
```

min(): İki sayıdan küçük olanını seçer.
min(x, y)

x: Birinci sayı.

y: İkinci sayı.

Örnek: İki analog girişten küçük olanı seçer. 800'ün üstündeki değerleri eler.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  int sensor1 = analogRead(A0);
  int sensor2 = analogRead(A1);
  int deger = min(sensor1, sensor2); // İki sensör değerinden küçük olanını seç.
  int sonuc = min(deger, 800); // Sonuç 800 veya altı olur.
  Serial.println(sonuc);
}
```

pow(): Sayının üssünü alır.

```
pow(taban,üs)
```

taban: Taban sayısı.

üs: Üs sayısı.

Örnek: 2^3 işleminin sonucunu verir.

```
sonuc = pow(2, 3); // Sonuç 8 olur.
```

sq(): Sayının karesini alır (Kendisiyle çarpar.).
sq(x)
x: Karesi alınacak sayı.

Örnek: 10^2 işleminin sonucunu verir.

```
sonuc = sq(10); // Sonuç 100 olur.
```

sqrt(): Sayının karekökünü verir.

sqrt(x)

x: Karekökü alınacak sayı.

Örnek: 16 işleminin sonucunu verir.

```
sonuc = sq(16); // Sonuç 4 olur.
```

random(): Rastgele sayı üretir. Long veri tipinde sayı döndürür.

```
random(max)
random(min, max)
```

min: Üretilecek sayının en küçük değeri (İsteğe bağlı.).

max: Üretilecek sayının en büyük değeri.

Örnek: Rastgele sayı üretir.

```
long rastgeleSayi;
void setup() {
  Serial.begin(9600);
}
void loop() {
  rastgeleSayi = random(300); // 0 - 299 arası rastgele sayı üret.
  Serial.println(rastgeleSayi);
  rastgeleSayi = random(10, 20); // 10 - 19 arası rastgele sayı üret.
  Serial.println(rastgeleSayi);
  delay(500);
}
```

randomSeed(): Yukarıdaki örnek seri port ekranını her kapatıp açmada aynı sırada rastgele sayı üretir. randomSeed() içine sabit bir sayı girildiğinde de sonuç aynıdır. Ancak randomSeed() her seferinde farklı bir değer alırsa üretilen rastgele sayı düzeni de farklı olacaktır. Parametre olarak boşta olan bir analog giriş seçilebilir. Analog girişler boştayken ortama göre farklı değerler alır.

`randomSeed(seed)`

seed: Rastgele sayı düzenini başlatacak sayı. Veri tipi unsigned long.

Örnek: Başlama düzenini analog girişten rastgele seçer.

```
long rastgeleSayi;
void setup() {
  Serial.begin(9600);
  randomSeed(analogRead(0));
}
void loop() {
  rastgeleSayi = random(300); // 0 - 299 arası rastgele sayı üret.
  Serial.println(rastgeleSayi);
  delay(500);
}
```

Örnek: LED'lerle rastgele animasyon programıdır.

```
byte i;
int sure = 100;
void setup() {
  for (i = 0; i <= 7; i++)  {
    pinMode(i, OUTPUT); // 0 - 7 pinleri çıkış yap.
  }
  randomSeed(analogRead(0));
}
void loop() {
  for (i = 0; i <= 7; i++) {
    digitalWrite(i, bitRead(random(256), i));
  }
  delay(sure);
}
```

Trigonometrik Fonksiyonlar

cos(): Açının kosinüs değerini radyan olarak verir. -1 ile 1 arasında double veri tipinde değer döndürür.

Örnek: Kosinüsün radyan değerini verir.

```
sonuc = cos(90); // Sonuc -0,45 olur.
```

sin(): Açının sinüs değerini radyan olarak verir. -1 ile 1 arasında double veri tipinde değer döndürür.

Örnek: Sinüsün radyan değerini verir.

```
sonuc = sin(90); // Sonuç 0,89 olur.
```

tan(): Açının tanjant değerini radyan olarak verir.

Örnek: Tanjantın radyan değerini verir.

```
sonuc = tan(90); // Sonuç -2 olur.
```

Örnek: Açı-radyan dönüşümlerini verir.

```
void setup() {
  Serial.begin(9600);
  Serial.println(cos(90)); // -0,45
  Serial.println(sin(90)); // 0,89
  Serial.println(tan(90)); // -2,00
}
void loop() {}
```

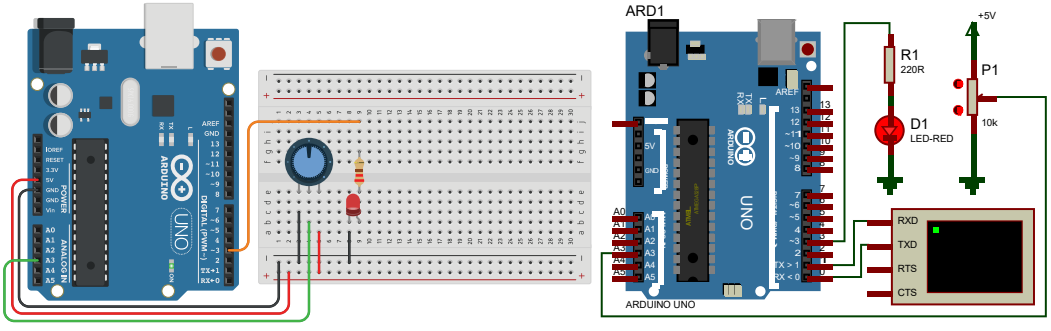
DİKKAT Matematiksel fonksiyonların içinde işlem yapılması yanlış sonuçlar vermesine neden olabilir.

```
abs(a++); // Bu kullanımdan kaçınılmalı. Yanlış sonuçlar verebilir.
// Bu kullanım tercih edilmelidir.
abs(a);
a++;  // İşlemler fonksiyon dışında yapılmalıdır.
```

Uygulama Adı: Analog Giriş-Çıkış Uygulaması — No.: 16

Amaç: Analog giriş-çıkış uygulaması yapmak.

Görsel 2.45'teki devrede potansiyometre 10 bitlik analog girişe bağlıyken LED 8 bitlik PWM çıkışına bağlıdır. Potansiyometreden alınan 0 -1023 arası değer doğrudan PWM çkışına verilirse (analogWrite(3, pot)) 8 bitlik PWM çıkışı potansiyometrenin bir tam dönüşünde dört tur dönecektir. map(pot, 0, 1023, 0, 255) fonksiyonu pot değişkeninden gelen 0 -1023 aralığını 0-255 aralığına eşler.



Görsel 2.45: Analog giriş-çıkış şema ve devresi

Analog giriş-çıkış uygulama programı aşağıdaki gibidir:

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  int pot = analogRead(A3);
  byte pwm = map(pot, 0, 1023, 0, 255);
  analogWrite(3, pwm);
  Serial.print("Analog giriş değeri = ");
  Serial.print(pot);
  Serial.print("\t Analog çıkış değeri = ");
  Serial.println(pwm);
}
```

İşlem Basamakları

1. Görsel 2.45'teki devreyi kurunuz. Programı yükleyiniz.
2. Potansiyometreyi baştan sona çevirerek LED'in durumunu ve seri port ekranını gözlemleyiniz.

SIRA SİZDE

1. analogWrite(3, pwm) komutunda **pwm** yerine pot yazınız. LED'in durumunu gözlemleyiniz.

Soru

1. Arduino'da PWM frekansı 980 Hz'den yüksek frekanslara ayarlanabilir mi? Araştırınız.

2.10. KESME İŞLEMLERİ

Kesme işlemi, mikrodenetleyicilerde önceden belirlenen görevleri ana programdan bağımsız şekilde kontrol ederek koşul sağlandığı anda programı durdurur ve belirlenen kesme fonksiyonun çalışmasını sağlar. Kesme fonksiyonundaki işlemler tamamlandığında program kaldığı yerden çalışmaya devam eder.

Arduino Uno kartında 2 adet dış kesme ve 3 adet zamanlayıcı kesmesi bulunmaktadır. Dış kesmeler Arduino UNO kartının 2 numaralı dijital pininde bulunan INT0 ve 3 numaralı dijital pininde bulunan INT1 kesmeleridir.

attachInterrupt() Fonksiyonu

Dış kesme kullanılırken kesme pini **dijital giriş** olarak ayarlanır. Kesmeyi başlatmak için setup() fonksiyonu içinde kesme fonksiyonu tanımlanır.

Kesme fonksiyonu tanımlama `attachInterrupt()` fonksiyonuyla yapılır. Haricî kesme kullanımındaki amaç ana programdaki işlemler devam ederken sensör vb. giriş bilgisini kaçırmamaktır.

Arduino UNO 'da kesme fonksiyonu için iki farklı söz dizimi mevcuttur.

```
attachInterrupt(kesmeNumarası, kesmeFonksiyonu, mod)
attachInterrupt(digitalPinToInterrupt(kesmePini), kesmeFonksiyonu, mod)
```

`kesmeNumarası:` Kesme numarası. Arduino Uno'da 2 numaralı pin için 0, 3 numaralı pin için 1 yazılır.

`kesmePini`: 2 girildiğinde digitalPinToInterrupt(kesmePini) fonksiyonu 0 döndürür, 3 girildiğinde 1 döndürür. Bu kullanımda kesme numarasıyla değil pin numarasıyla işlem yapılır.

`kesmeFonksiyonu:` Kesme oluştuğunda çağrılan fonksiyon. Parametre almaz. Değer döndürmez.

`mod:` Kesmenin tetiklenme modudur. Dört çeşittir.

LOW: Pindeki voltaj 0 V'sa kesme oluşur.

CHANGE: Pinde oluşacak voltaj değişimlerinde kesme gerçekleşir.

RISING: Yükselen kenarlarda kesme gerçekleşir. Pindeki voltaj değeri 0 V'tan 5 V'a çıkarken kesme gerçekleşir.

FALLING: Düşen kenarlarda kesme gerçekleşir. Pindeki voltaj değeri 5 V'tan 0 V'a düşerken kesme gerçekleşir.

Örnek: Ana programda blink uygulaması devam ederken butona basıldığında seri port ekranına "Butona basıldı." yazdırılmaktadır. Benzer işlem ana programda if kullanılarak da yapılabilir. Ancak butona basma anı if dışındaki komutlar icra edildiği zamana gelirse işlem gerçekleşmez veya gecikmeli olarak gerçekleşir.

```
const byte LED = 13;
const byte kesmePini = 2;
void setup() {
  Serial.begin(9600);
  pinMode(LED, OUTPUT);
```

```
   pinMode(kesmePini, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(kesmePini), kesmeFonksiyonu, FALLING);
}
void loop() { // Blink uygulaması
  digitalWrite(LED, HIGH);
  delay(1000);
  digitalWrite(LED, LOW);
  delay(1000);
}

void kesmeFonksiyonu() { // Kesme gerçekleştiğinde buradaki işlemler yapılır.
  Serial.println("Butona basıldı");
}
```

Örnek: Butona üç kez basıldığında kesme aktif edilmektedir.

```
const byte dahiliLED = 13;
const byte hariciLED = 12;
const byte kesmePini = 2; //Butona bağlı.
byte sayac;
void setup() {
  Serial.begin(9600);
  pinMode(dahiliLED, OUTPUT);
  pinMode(hariciLED, OUTPUT);
  pinMode(kesmePini, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(kesmePini), kesmeFonksiyonu, FALLING);
}

void loop() { //Blink uygulaması
  digitalWrite(dahiliLED, HIGH);
  delay(1000);
  digitalWrite(dahiliLED, LOW);
  delay(1000);

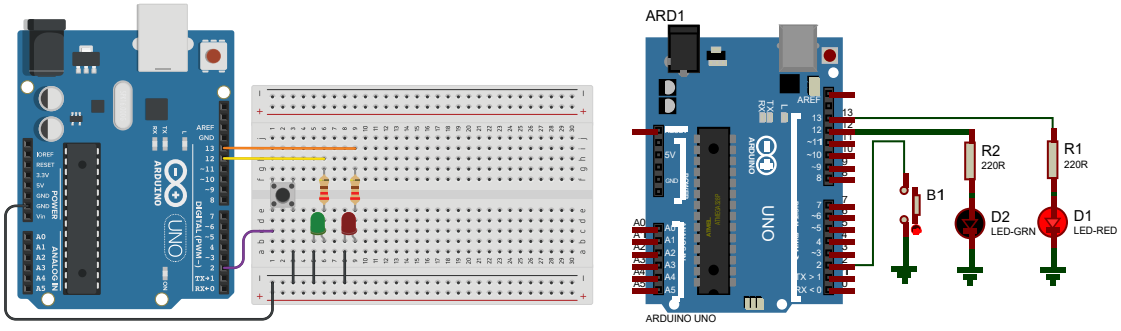
  digitalWrite (hariciLED, LOW);//Harici LED'i söndür.
  if (sayac > 3)
    detachInterrupt(digitalPinToInterrupt(kesmePini));
}

void kesmeFonksiyonu() { //Kesme gerçekleştiğinde buradaki işlemleri yap.
  digitalWrite (hariciLED, HIGH); //Harici LED'i yak.
  sayac++;
}
```

Uygulama Adı: Kesme Uygulaması — No.: 17

Amaç: **Kesme uygulaması yapmak.**

2.46'daki programda ana döngüde blink uygulaması çalışırken butona basıldığında kesme aktif edilmekte ve haricî LED anında yanmaktadır. Haricî LED'in sönme komutu ise ana döngüde verildiğinden sönme işlemi ana programda icra edilen komut satırına bağlı olarak 0 ile 2 saniye arasında süre alır.



Görsel 2.46: Kesme uygulaması şema ve devresi

Kesme uygulama programı aşağidaki gibidir:

```
const byte dahiliLED = 13;
const byte hariciLED = 12;
const byte kesmePini = 2; // Butona bağlı.
void setup() {
  Serial.begin(9600);
  pinMode(dahiliLED, OUTPUT);
  pinMode(hariciLED, OUTPUT);
  pinMode(kesmePini, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(kesmePini), kesmeFonksiyonu, FALLING);
}
void loop() { // Blink uygulaması
  digitalWrite(dahiliLED, HIGH);
  delay(1000);
  digitalWrite(dahiliLED, LOW);
  delay(1000);
  digitalWrite (hariciLED, LOW);// Haricî LED'i söndür.
}
```

İşlem Basamakları

1. Görsel 2.46'daki devreyi kurunuz. Programı yükleyiniz.
2. Blink uygulaması çalışırken butona basınız. LED'lerin çalışmasını gözlemleyiniz.

SIRA SİZDE

1. Haricî kesme tetiklendiğinde seri port ekranına "Butona basıldı." yazdırınız.

Soru

1. Haricî kesme tetiklendiğinde loop döngüsü içindeki komutlar niçin icra edilmemektedir? Açıklayınız.

2. Aşağıdaki programla yazdığınız programın farkı nedir?

```
const byte dahiliLED = 13;
const byte hariciLED = 12;
const byte kesmePini = 2; //Butona bağlı.
volatile bool a = 1;

void setup() {
  Serial.begin(9600);
  pinMode(dahiliLED, OUTPUT);
  pinMode(hariciLED, OUTPUT);
  pinMode(kesmePini, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(kesmePini), kesmeFonksiyonu, FALLING);
}

void loop() { //Blink uygulaması
  digitalWrite(dahiliLED, HIGH);
  delay(1000);
  digitalWrite(dahiliLED, LOW);
  delay(1000);
}

void kesmeFonksiyonu() { //Kesme gerçekleştiğinde buradaki işlemleri yap.
  delay(100);
  if (a)
    digitalWrite (hariciLED, HIGH); //Harici LED'i yak.
  else  digitalWrite (hariciLED, LOW);//Harici LED'i söndür.
  a = !a;
}
```

detachInterrupt() Fonksiyonu

Belirtilen kesmeyi sonlandırır.

`detachInterrupt(kesmeNumarası)`

`detachInterrupt(digitalPinToInterrupt(kesmePini))`

kesmeNumarası: Kesme numarası. Arduino Uno 'da 2 numaralı pin için 0, 3 numaralı pin için 1 yazılır.
kesmePini: Arduino Uno 'da 2 veya 3 numaralı pinlerdir.

Aşağıdaki programda ana döngüde blink uygulaması çalışırken butona basıldığında kesme aktif edilerek haricî LED yanmaktadır. Kesme üç kez çalıştıktan sonra sonlandırılmıştır.

```
const byte hariciLED = 12;
const byte kesmePini = 2; // Butona bağlı.
byte sayac;
void setup() {
  Serial.begin(9600);
  pinMode(dahiliLED, OUTPUT);
  pinMode(hariciLED, OUTPUT);
  pinMode(kesmePini, INPUT_PULLUP); // 2 numaralı giriş 0 olduğunda kesme aktif olur.
  attachInterrupt(digitalPinToInterrupt(kesmePini), kesmeFonksiyonu, FALLING);
}
void loop() { // Blink uygulaması
  digitalWrite(dahiliLED, HIGH);
  delay(1000);
  digitalWrite(dahiliLED, LOW);
  delay(1000);

  digitalWrite (hariciLED, LOW);// Haricî LED'i söndür.
  if (sayac > 3)
    detachInterrupt(digitalPinToInterrupt(kesmePini)); // 2 numaralı pindeki kesmeyi sonlandır.
}
void kesmeFonksiyonu() { // Kesme gerçekleştiğinde buradaki işlemleri yap.
  digitalWrite (hariciLED, HIGH); // Haricî LED'i yak.
  delay(1000); // Harici LED'i 1 saniye yak.
  sayac++; // Sayaç değişkenini 1 artır.
}
```

Zamanlayıcı Kesmeleri

Zaman kesmesi (timer interrupt), belirli zaman aralıklarında istenilen görevleri ana programdan bağımsız gerçekleştirmek için kullanılır. Blink, kara şimşek vb. uygulamalar zaman kesmesiyle çalıştırılırken ana döngüde farklı bir program çalışmaya devam eder.

Arduino'da bulunan 16 MHz'lik işlemci, sayaç kaydedicisini [counter register (kauntır recîstır)] her saat darbesinde (clock) arttırır. Frekans bölücü (prescalar) kullanılarak daha düşük saat frekansları elde edilir. Uno'da üç adet zamanlayıcı bulunur. delay() ve millis() fonksiyonlarının da kullanıldığı Timer0 8 bit (255), servo kütüphanesinin kullandığı Timer1 16 bit (65535), tone() fonksiyonunun kullandığı Timer2 8 **bit**tir. Zamanlayıcılar her saat darbesinde bir artarak belirli

sayılara ulaşınca taşma (overflow) gerçekleşir. Taşma değerine ulaştığında zamanlayıcı sıfırlanır ve kesme oluşur. Bu kesme ana programdaki işlemleri durdurarak ISR(TIMERx_OVF_vect) fonksiyonu içindeki komutların icra edilmesini sağlar. TIMERx ifadesinde x yerine 0, 1, 2'den biri yazılarak ilgili timer seçilir. Taşma modu haricinde karşılaştırma modu da mevcuttur.

Karşılaştırma zamanlayıcısında taşma oluşmasını (255 veya 65535 sayılarını) beklemeksizin istenen sayıya ulaşıldığında kesme işlemi gerçekleşir ve ISR(TIMERx_COMPA_vect) fonksiyonu içindeki komutlar çalışır. Karşılaştırma değeri

$$\text{Karşılaştırma değeri} = \left(\frac{\text{İşlemci frekansı}}{\text{Frekans} * \text{Ölçek}}\right) - 1$$

formülüyle hesaplanır. Elde etmek istenen frekans yerine 1/T yazılarak saniye cinsinden süre de yazılabilir. Zamanlayıcı kesmesini karşılaştırma modunda çalıştırmak için 8 **bit**lik **TCCR1A**, **TCCR1B**, **TIMSK1** ve 16 **bit**lik **OCR1A** kaydedicilerin ayarlanması gerekir.

Zamanlayıcı Kesmesini Karşılaştırma Modunda Ayarlama

TCCR1A Kaydedicisi

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
COM1A1	COM1A0	COM1B1	COM1AB0				

Karşılaştırma modunu aktif yapmak için TCCR1A kaydedicisi 0 yapılır (TCCR1A = 0).

TCCR1B Kaydedicisi

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
					CS12	CS11	CS10

TCCR1B kaydedicisinin ilk üç **bit**i ölçeği belirler. Tablo 2.9'da ölçeklendirme seçenekleri yer almaktadır.

Tablo 2.9: Clock Sinyalini Ölçeklendirme (Prescalar)

CS12	CS11	CS10	Açıklama
0	0	0	Zamanlayıcı/Sayıcı kapalı
0	0	1	Clock/1 (16 MHz)
0	1	0	Clock/8
0	1	1	Clock/64
1	0	0	Clock/256
1	0	1	Clock/1024
1	1	0	T1 pinden haricî clock sinyali. Düşen kenar tetikleme.
1	1	1	T1 pinden haricî clock sinyali. Yükselen kenar tetikleme.

Öncelikle TCCR1B kaydedicisi sıfırlanır. (TCCR1B = 0) Örneğin 256 ölçek için TCCR1B |= B00000100 binary sayısıyla **bit**sel veya (|) işlemi yapılarak kurulum yapılır. (Aynı kurulum TCCR1B |= (1 << CS12) şeklinde de yapılabilir. **Bit** numarasını içinde barındıran CS12 sabiti (içeriği 2) lojik 1'i iki **bit** sola kaydırarak **bit**sel veya (|) işlemine tabi tutulur.)

TIMSK1 Kaydedicisi

BIT	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
						OCIE1B	OCIE1A	TOIE1

Taşma zamanlayıcısı kesmesini aktifleştirmek için TOIE1 **bit**i (0. Bit), karşılaştırma zamanlayıcısı kesmelerini aktifleştirmek için OCIE1A ve OCIE1B **bit**leri 1 yapılır.

Son olarak 16 **bit**lik OCR1A kaydedicisine karşılaştırma değeri yüklenir (OCR1A = 62499).

Setup fonksiyonunda zamanlayıcı kesmesinin ayarları yapılmadan önce cli() fonksiyonuyla kesmeler durdurulur. Ayarlar yapıldıktan sonra sei() fonksiyonuyla kesmeler aktif edilir.

Örnek: Blink uygulamasının zamanlayıcı kesmesiyle karşılaştırma modunda çalıştırılır.

```
void setup() {
  pinMode(13, OUTPUT);
pinMode(12, OUTPUT);
  cli();                        // Ayarlar yapılana kadar kesmeleri durdur.
  TCCR1A = 0;                   // Karşılaştırıcı çıkış modunu aktif yap.
  TCCR1B = 0;                   // TCCR1B'yi sıfırla.
  TCCR1B |= B00000100;          // CS12'yi 1 yap. Ölçek 256 ayarlandı.
  TIMSK1 |= B00000010;          // OCIE1A'yı 1 yap.

  // 1 Hz 256 ölçek ile:  Karşılaştırma değeri = [16000000 / (256 * 1)] - 1 = 62499
  OCR1A = 62499;                // Karşılatırma kaydedicisini 62499'e ayarla.
  sei();                        // Kesmeleri aktif yap.
}

void loop() {// Zamanlayıcı kesmesi blink uygulmasını çalıştırırken buraya yazılacak kodlar da
çalışır.


digitalWrite(12, !digitalRead(12)); // 12 No.lu pin çıkışı değiştir.
}

// Karşılaştırıcı kesme fonksiyonu saniyede bir kez tetiklenecek şekilde setup fonksiyonu içinde
ayarlandı.
ISR(TIMER1_COMPA_vect) {
  digitalWrite(13, !digitalRead(13)); // 13 No.lu pin çıkışı değiştir.
}
```

2.11. EEPROM İŞLEMLERİ

Arduino kartlar üzerindeki mikrodenetleyicide EEPROM bulunur. EEPROM kartın enerjisi kesildiğinde (küçük bir sabit sürücü gibi) değerleri tutulan bellektir. EEPROM.h kütüphanesi, **byte**ları EEPROM hafızaya yazmayı ve buradan okumayı sağlar. Farklı kartlar farklı EEPROM hafıza alanına sahiptir. Uno'da EEPROM hafıza 1 kB'tır. EEPROM belleğinin belirli bir (100,000 kez) yazma ömrü vardır.

2.11.1. EEPROM Yazma İşlemleri

EEPROM.write() Fonksiyonu

EEPROM'da belirtilen adrese 1 **byte** veri yazar. Değer döndürmez.

`EEPROM.write(adres, deger)`

`adres:` Verinin yazılacağı adres. Veri tipi int (1 kB hafıza için 0-1023 arası adres.).

`deger:` 1 **byte**lık veri. Veri tipi **byte** (0-255 arası değer.).

Örnek: Program A0 analog girişinden (pot, sensör vb.) aldığı bilgileri tüm hafızaya sırayla yazmaktadır.

```
#include <EEPROM.h>
byte deger;
void setup() {
  int hafizaBoyutu = EEPROM.length(); // Kartın hafıza boyutunu al.
  for (int adres = 0; adres < hafizaBoyutu; adres++) {
    int deger = analogRead(A0) / 4; // 0 -1023 arası gelen veriyi 0 - 255 arası değer olarak
ata.
    EEPROM.write(adres, deger); // Analog veriyi tüm hafıza sırayla kaydet.
    delay(10); // 10 milisaniye bekle.1 kB hafıza yaklaşık 20 saniyede dolar.
  }
}
void loop() {
  /*100.000 yazma/okuma ömrünü daha uzun süre kullanabilmek için
    program loop döngüsüne yazılmamıştır.*/
}
// Yazılan değerleri görmek için EEPROM.read örneğini yükleyin.
```

2.11.2. EEPROM Okuma İşlemleri

EEPROM.read() Fonksiyonu

EEPROM'dan bir **byte** okur. Girilen adresteki **byte**ı döndürür.

`EEPROM.read(adres)`

`adres:` Verinin okunacağı adres. Veri tipi int.

Örnek: Program tüm EEPROM hafızasını sırayla seri port ekranına yazmaktadır.

```
#include <EEPROM.h>
byte deger;
void setup() {
  Serial.begin(9600);
  int hafizaBoyutu = EEPROM.length(); // Kartın hafıza boyutunu al.
  for (int adres = 0; adres < hafizaBoyutu; adres++) {
    deger = EEPROM.read(adres); // Analog veriyi 10 ms aralıklarla tüm hafızadan sırayla oku.
    Serial.print("Adres = ");
```

EEPROM.read() Fonksiyonu

EEPROM'dan bir **byte** okur. Girilen adresteki **byte**ı döndürür.

`EEPROM.read(adres)`

adres: Verinin okunacağı adres. Veri tipi int.

Örnek: Program tüm EEPROM hafızasını sırayla seri port ekranına yazmaktadır.

```
#include <EEPROM.h>
byte deger;
void setup() {
  Serial.begin(9600);
  int hafizaBoyutu = EEPROM.length(); // Kartın hafıza boyutunu al.
  for (int adres = 0; adres < hafizaBoyutu; adres++) {
    deger = EEPROM.read(adres); // Analog veriyi 10 ms aralıklarla tüm hafızadan sı-
rayla oku.
    Serial.print("Adres = ");
    Serial.print(adres);
    Serial.print("\t Değer = ");
    Serial.println(deger);
    delay(10);
  }
}
void loop() {}
```

NOT EEPROM[] ifadesi bir dizi gibi EEPROM hücrelerine ulaşmayı sağlar.

```
x = EEPROM[0]; // İlk EEPROM hücresini oku.
 EEPROM[0] = x; // İlk EEPROM hücresine yaz.
```

EEPROM.read() Fonksiyonu

EEPROM.read() ve EEPROM.write() fonksiyonlarının birleşiminden oluşmuştur. Adresteki değeri önce okur, yeni yazılacak değer kaydedilmiş olandan farklıysa yazma işlemi gerçekleştirilir. Böylelikle yazma ömründen tasarruf edilir.

`EEPROM.update(adres, deger)`

adres: Verinin güncelleneceği adres. Veri tipi int.

deger: 1 **byte**lık veri. Veri tipi **byte**.

Örnek: EEPROM.update kullanımını gösterir.

```
#include <EEPROM.h>
void setup(){
  for (int i = 0; i < 255; i++) {
    EEPROM.update(i, i);
  }
  for (int i = 0; i < 255; i++) {
    // 3 numaralı adrese 12 bilgisi bir kez yazılır. Diğer 254 tekrarda işlem yapılmaz.
```

```
    EEPROM.update(3, 12);
  }
}
```

EEPROM.put() Fonksiyonu

EEPROM'a herhangi bir veri tipini yazmak için kullanılır.

`EEPROM.put(adres, deger)`

adres: Verinin yazılacağı adres. Veri tipi int.

deger: Herhangi bir veri tipindeki değer.

Örnek: EEPROM hafızaya veri kaydeder.

```
#include <EEPROM.h>
int adres = 0;
struct urun {
  String ad;
  float agirlik;
  byte uzunluk;
};
void setup() {
  Serial.begin(9600);
  urun urun1 = { // Kaydedilecek veriler.
    "Ürün1",
    57.29,
    168
  };
  EEPROM.put(adres, urun1);
  Serial.print("Veriler sonraki örnekte EEPROM.get() fonksiyonuyla okunacaktır.");
}
void loop() {}
```

EEPROM.get() Fonksiyonu

EEPROM'dan herhangi bir veri tipini okumak için kullanılır.

`EEPROM.get(adres, ozelNesne)`

adres: Verinin okunacağı adres. Veri tipi int.

ozelNesne: EEPROM'dan okunan farklı tiplerdeki verilerin atanacağı, önceden oluşturulmuş nesne.

Örnek: EEPROM hafızadan veri okur.

```
#include <EEPROM.h>
int adres = 0;
struct urun { // Ürün isimli nesne oluşturuldu.
  String ad;
  float agirlik;
  byte uzunluk;
```

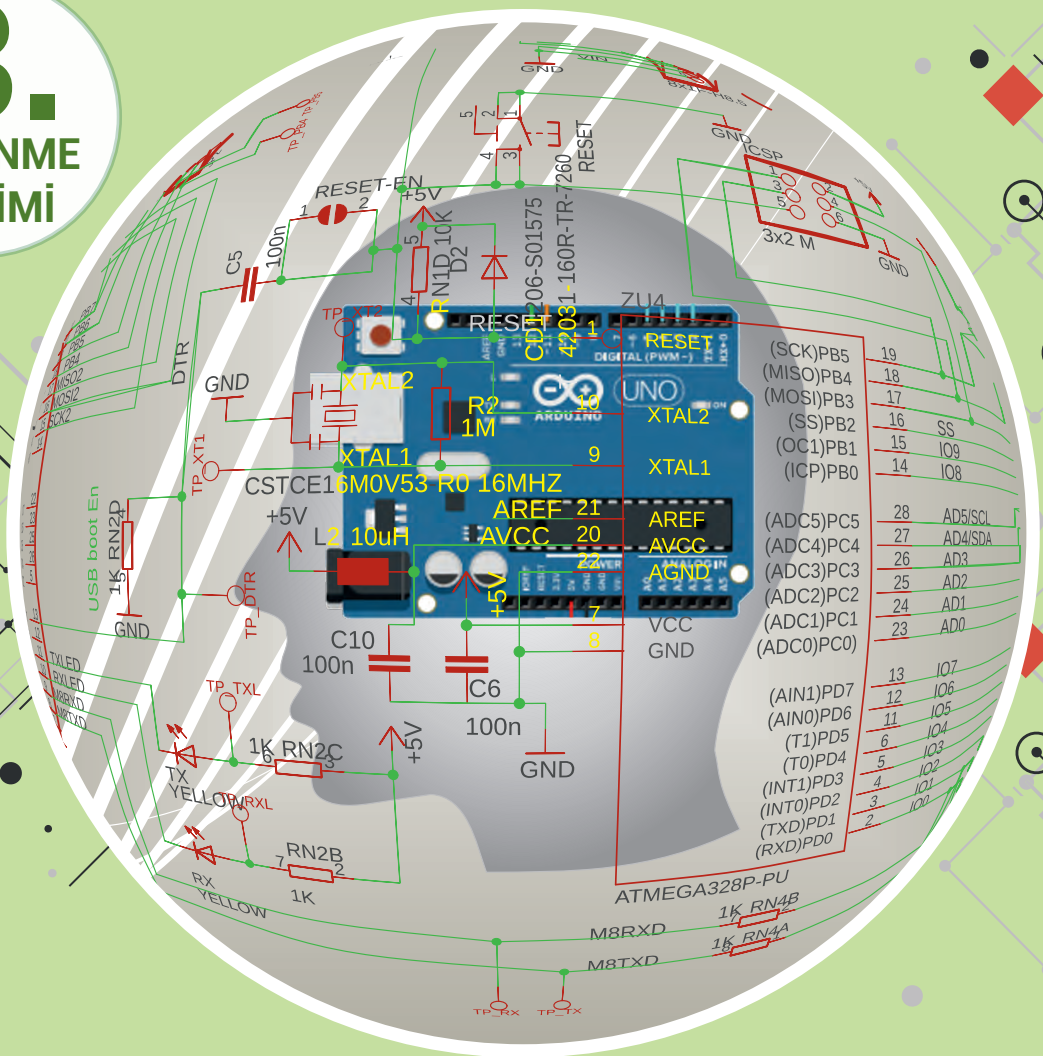
```
};
void setup() {
  Serial.begin(9600);
  urun urun1; // ÜrÜn nesnesiyle ürün1 isimli değişken oluşturuldu.
  EEPROM.get(adres, urun1); // Ürün1'e ait bilgileri hafızadan al.
  Serial.println(urun1.ad);
  Serial.println(urun1.agirlik);
  Serial.println(urun1.uzunluk);
}
void loop() {}
```

Örnek: Tüm EEPROM hafızayı okur veya siler.

```
#include <EEPROM.h>
void setup() {
  Serial.begin(9600);
  eepromOku();
  //eepromReset();
}
void eepromOku() {
  int hafizaBoyutu = EEPROM.length(); // Kartın hafıza boyutunu al.
  for (int adres = 0; adres < hafizaBoyutu; adres++) {
    byte deger = EEPROM.read(adres);

    Serial.print(adres);
    Serial.print(" ");
    Serial.println(deger);
    delay(1);
  }
}
void eepromReset() {
  Serial.println("Hafıza sıfırlanıyor. Lütfen bekleyiniz...");
  int hafizaBoyutu = EEPROM.length(); // Kartın hafıza boyutunu al.
  for (int adres = 0; adres < hafizaBoyutu; adres++) {
    EEPROM.write(adres, 0); // Tüm hafızayı sıfırla.
  }
  Serial.println("Tüm hafıza silindi.");
}
void loop() {
}
```

3. ÖĞRENME BİRİMİ



MİKRODENETLEYİCİ UYGULAMALARI

NELER ÖĞRENECEKSİNİZ ?

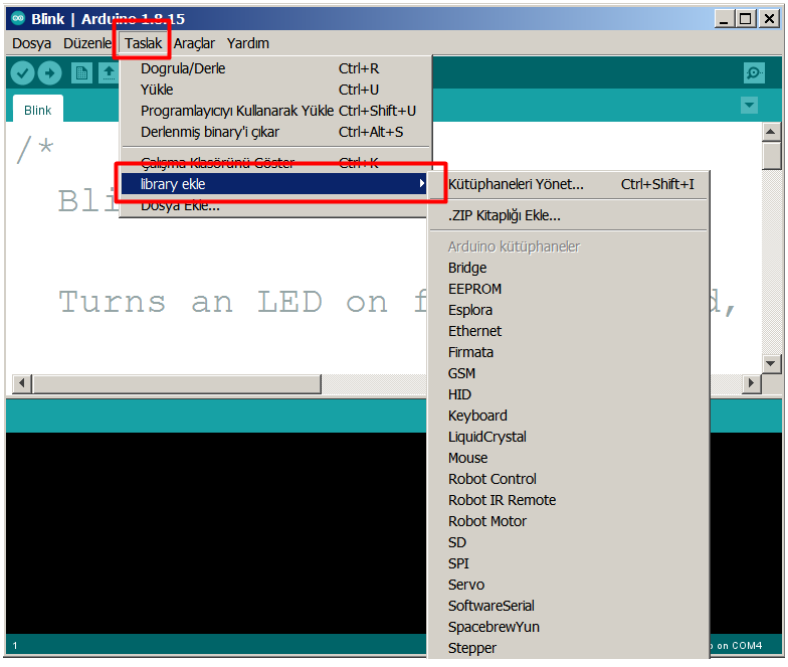
- Kütüphane Dosyası Yükleme
- Keypad Uygulamaları
- LCD Ekran Uygulamaları
- Sensör Uygulamaları
- Elektrik Motor Uygulamaları
- Haberleşme Uygulamaları

TEMEL KAVRAMLAR

kütüphane
haberleşme
radyo frekans
nesnelerin interneti
aplikasyon
sensör
motor
PWM
robot
PİD

3.1. KÜTÜPHANE DOSYASI YÜKLEME

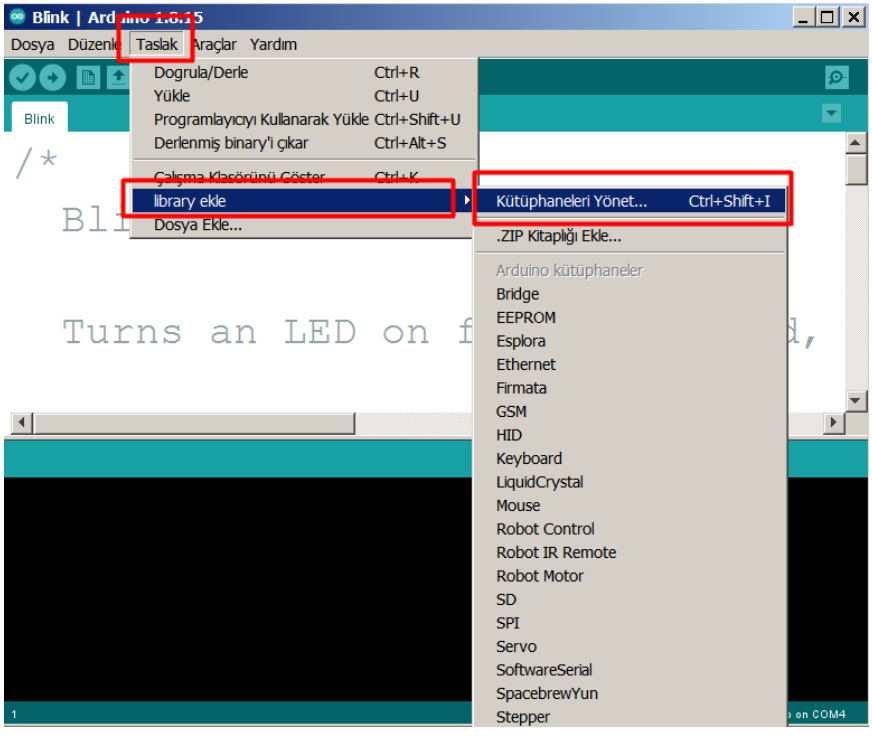
Arduino ortamı, çoğu programlama platformunda olduğu gibi kütüphaneler kullanılarak genişletilebilir. Kütüphaneler hazır fonksiyonlarıyla pek çok işlemi kolaylaştırır. Kütüphaneler fonksiyonların dosyalanmış şekli olarak da düşünülebilir. Örneğin dâhilî Liquid Crystal kütüphanesi LCD ekrana kolayca karakter yazmaya yardımcı olur. Bir taslağa kütüphane eklemek için Görsel 3.1'de görüldüğü gibi "Taslak" menüsünden "library ekle" komutu seçilir. Arduino IDE ile beraber bazı kütüphaneler gelmektedir. Bunun yanında kütüphane yöneticisiyle yeni kütüphaneler internetten indirilebilir veya bir kütüphane yazılabilir.



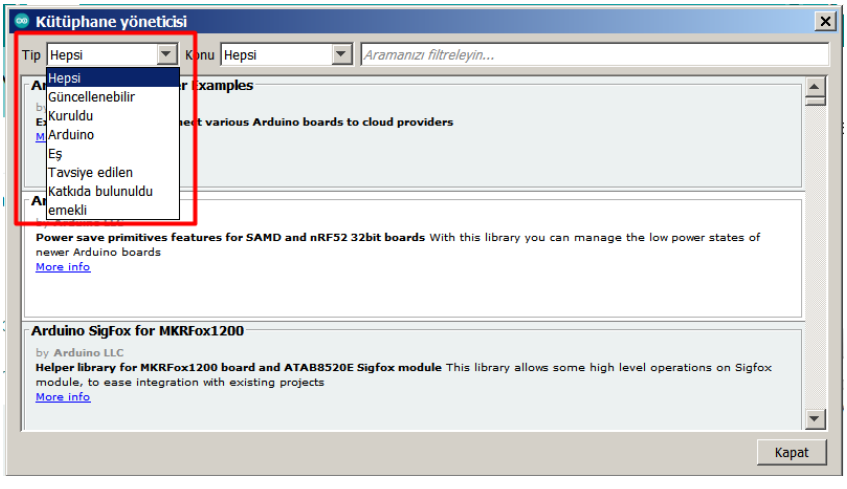
Görsel 3.1: Kütüphane ekleme

İlave Kütüphane Yükleme

Yeni bir kütüphane yüklemek için Görsel 3.2'de görüldüğü gibi "Taslak → Library Ekle →Kütüphaneleri Yönet" yolu seçilir. Kütüphane yöneticisi açıldığında yüklü kütüphaneler ile yüklenebilecek kütüphanelerin listesi görülür (Görsel 3.3). Yüklemek istenilen kütüphanenin üzerindeki "kur" düğmesine basıldığında kütüphane yüklenir. Artık üzerinde "yüklendi" ifadesi yer alır. "Taslak → Library Ekle" menüsünde yüklenen yeni kütüphane görülür.



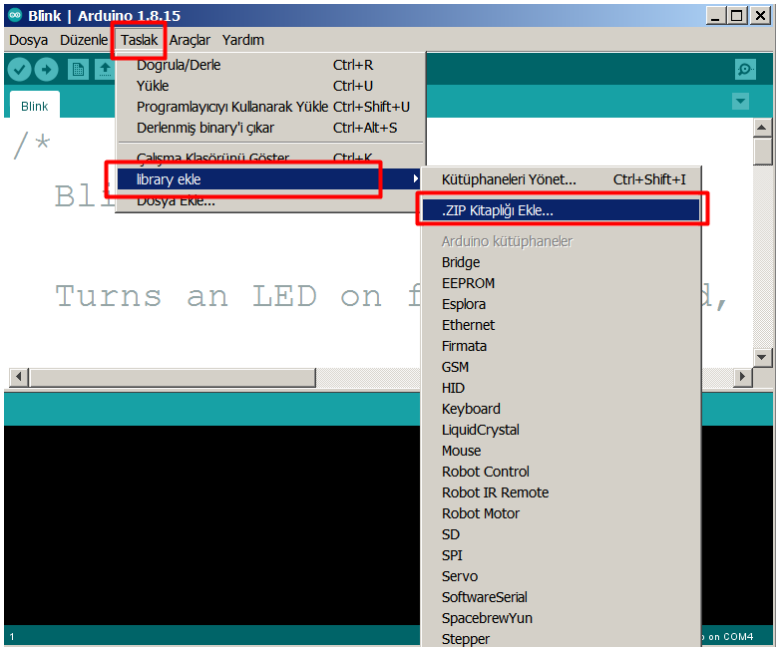
Görsel 3.2: Yeni kütüphane ekleme



Görsel 3.3: Kütüphane yöneticisi

Sıkıştırılmış Dosya (.zip) Kitaplığını İçe Aktarma

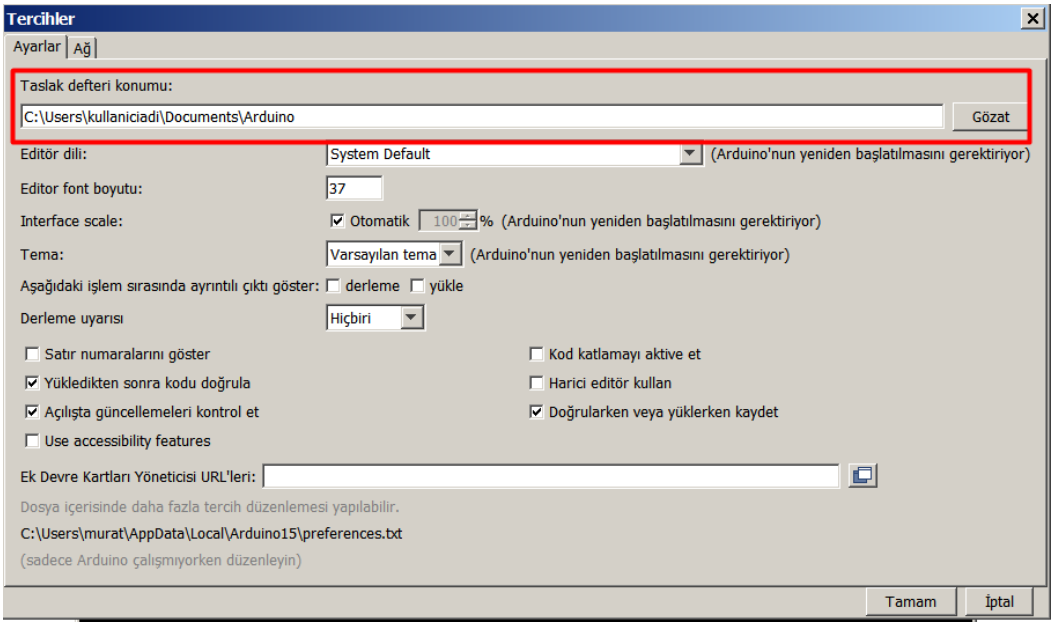
Kütüphaneler genellikle bir **.zip** dosyası veya klasörü olarak **github.com** vb. web sitelerinde dağıtılır. Klasörün adı genelde kütüphanenin adıdır. Klasörün içinde bir **.cpp** dosyası, bir **.h** dosyası ve genellikle bir **keywords.txt** dosyası, örnekler klasörü ve kütüphanenin gerektirdiği diğer dosyalar bulunur. Sıkıştırılmış dosyayı (.zip) içe aktarmak için Görsel 3.4'te "Taslak → Library Ekle → .ZIP Kitaplığı Ekle" seçilir. Dosya yüklendiğinde "Taslak → Library Ekle" menüsünde yüklenen yeni kütüphane görülür.



Görsel 3.4: ZIP kitaplığı ekleme

Kütüphane Dosyalarını Elle Yükleme

Taslak klasörünün konumuna "Dosya→ Tercihler → Taslak Defteri Konumu" bölümünden ulaşılır ve bu konum istenirse değiştirilebilir (Görsel 3.5). Daha sonra kütüphanenin **.zip** dosyasının indirildiği dizine gidilir. **.zip** dosyası tüm klasör yapısıyla birlikte geçici bir klasöre çıkarılır, ardından kütüphanenin adına sahip olması gereken ana klasör seçilerek "C:\Users\kullaniciadi\Documents\Arduino" klasörüne kopyalanır. Daha sonra "Taslak → Library Ekle" menüsünde yüklenen yeni kütüphane kontrol edilir.



Görsel 3.5: Taslak defteri konumu

Arduino Web Editörde Kütüphaneler

Taslak web editöre (https://create.arduino.cc/editor) yüklendiğinde veya web editöre de yazıldığında Arduino IDE kütüphane yöneticisinde yer alan tüm kütüphaneler kurulu olduğundan kütüphane kurulumu gerekmez. Ancak Arduino kütüphane yöneticisinde bulunmayan farklı bir yerden edinilmiş kütüphane dosyaları web editöre yüklenerek kullanılır (Görsel 3.6). Web editörde tüm kütüphanelerin son (güncel) versiyonu yer aldığından eski versiyon kütüphane komutlarıyla yazılmış taslaklar derleme sırasında hata verir.



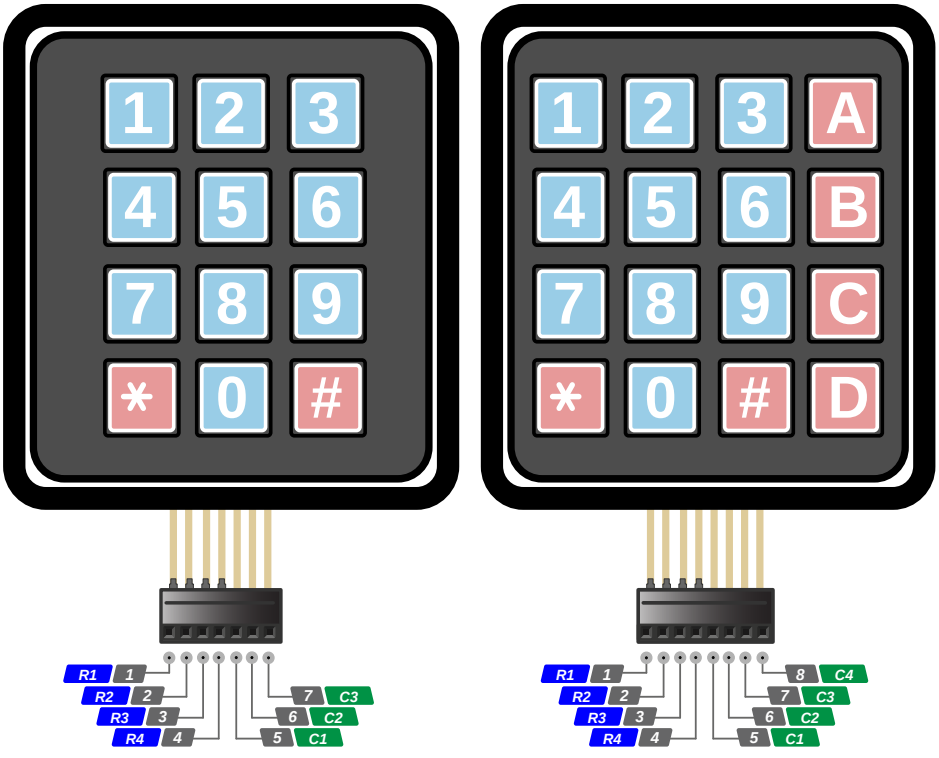
Görsel 3.6: Web editörde kütüphaneler

DİKKAT

Kitapta yer alan kütüphanelerin versiyonları kütüphane komutunun yanında yazmaktadır. Bazı kütüphanelerin güncel versiyonlarında fonksiyonların yapıları tamamen değiştiğinden derleyici hata verebilir. Kitapta yazılmış programların ilk satırlarında belirtilen kütüphane versiyonları kullanılmalı veya yeni versiyona göre taslak yeniden düzenlenmelidir.

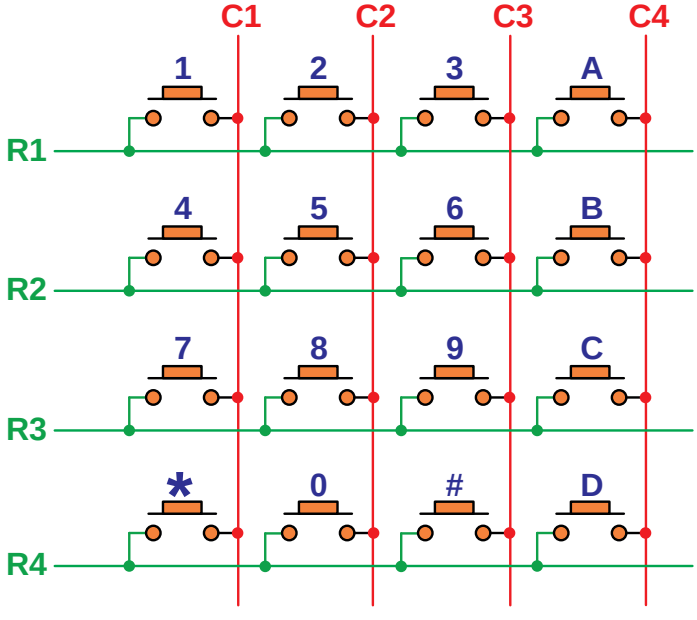
3.2. KEYPAD UYGULAMALARI

Tuş takımı, veri girişi gerektiren projelerde kullanılır. Tuş takımı bir buton grubudur. Satır ve sütunlardan oluşan buton gruplarının push buton (keyboard) veya membran (keypad) gibi farklı malzemelerden üretilmiş çeşitleri mevcuttur. Görsel 3.7'de 4x3 ve 4x4 membran tuş takımları girilmektedir.



Görsel 3.7: 4x3 ve 4x4 membran tuş takımı pinleri

Görsel 3.8'de 4x4 tuş takımının iç yapısı ve bağlantıları verilmiştir. Butonlardan birine basıldığında buton, bağlantılı olduğu satır ve sütun hattını birleştirir. Her buton için bir satır ve bir buton bilgisi aktif olur. Bir numaralı butona basıldığında R1: C1, iki numaralı butona basıldığında R1: C2 vb. şeklinde 16 farklı seçenek oluşur.



Görsel 3.8: 4x4 tuş takımının iç yapısı ve bağlantıları

Uygulama Adı: Keypad Uygulaması — No.: 1

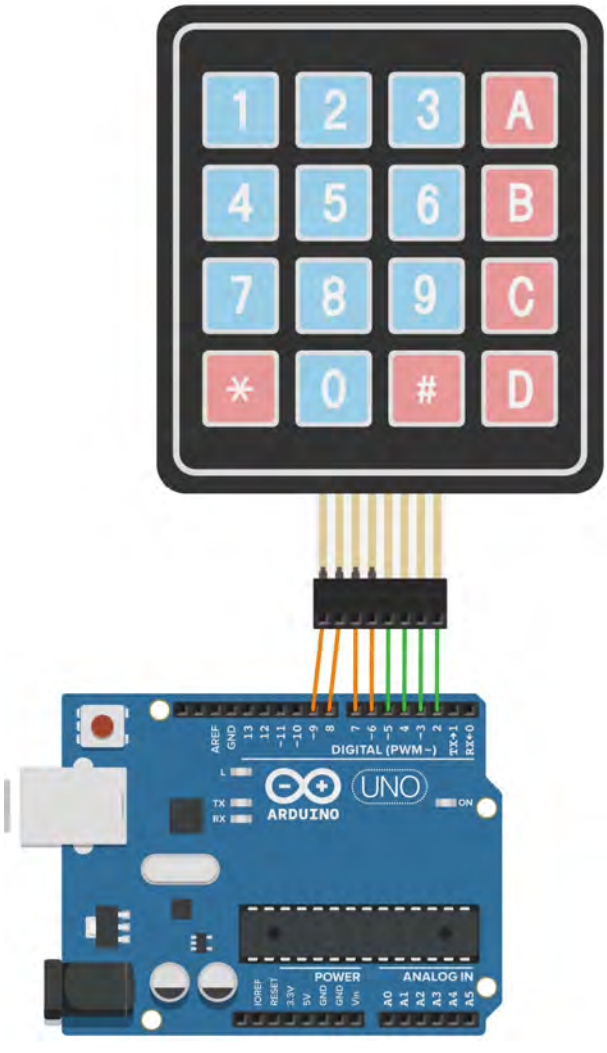
Amaç: Keypad uygulaması yapmak.

Görsel 3.9'daki devrede tuş takımı uygulaması verilmiştir. Keypad kütüphanesini kurmak için menülerden "Taslak → Library Ekle → Kütüphaneleri Yönet" seçeneğinden arama kutusuna "Keypad" yazılıp Keypad (Keypad by Mark Stanley and Alexander Brevig) kütüphanesi bulunarak kurulur. Daha sonra "Taslak → Library Ekle → Keypad" seçilerek programın başına eklenir. Bu programın bir örneğine menülerden "Dosya → Örnekler → Keypad → CustomKeypad" adımları izlenerek de ulaşılır.

Kullanılan tuş takımının satır ve sütun bilgileri, tuş dizilimi ve Arduino'ya bağlanan pinlerin bilgileri belirlenen değişkenlere atanır. Daha sonra kütüphane komutuyla tuş haritası oluşturulur.

```
Keypad tusTakimi = Keypad( makeKeymap(tusDizilimi), satirPinleri, sutunPinleri, satir, sutun);
```

Oluşturulan "`tusTakimi`" nesnesiyle kütüphane fonksiyonları çağrılır. `tusTakimi.getKey()` fonksiyonu, basılan tuşa ait karakteri döndürür.



Görsel 3.9: Tuş takımı uygulaması

Tuş takımı uygulaması programı aşağıdaki gibidir:

```
#include <Keypad.h> //Tuş takımı kütüphanesi. V3.1.1
const byte satir = 4; //Dört satır.
const byte sutun = 4; //Dört sütun.
char tusDizilimi[satir][sutun] = { //Tuş takımındaki sembollerin dizilimini tanımlama.
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
byte satirPinleri[satir] = {9, 8, 7, 6};
//{R1, R2, R3, R4} Satır pinlerinin Arduino pinlerine bağlantısı.
byte sutunPinleri[sutun] = {5, 4, 3, 2};
//{C1, C2, C3, C4} Sütun pinlerinin Arduino pinlerine bağlantısı.
//tusTakimi adında nesne oluşturma.
Keypad tusTakimi = Keypad( makeKeymap(tusDizilimi), satirPinleri, sutunPinleri, satir, sutun);
void setup() {
  Serial.begin(9600);
}
void loop() {
  char tus = tusTakimi.getKey(); //Basılan tuşu oku.
  if (tus) {
  Serial.println(tus); //Basılan tuşu seri ekrana yaz.
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.9'daki devreyi kurunuz.
4. Taslaktaki tuş takımı dizilimini elinizdeki tuş takımına göre düzenleyiniz.
5. Programı yükleyiniz.
6. Basılan tuşu seri port ekranında gözlemleyiniz.
7. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Doğru şifre girildiğinde kapı kilidini açan uygulamayı gerçekleştiriniz.

```
#include <Keypad.h> //Tuş takımı kütüphanesi.
```

```
byte hane = 0, LED = 13; //Selenoid kapı kilidi bağlanabilir.
char sifre[] = "123ABC"; // Kullanılacak şifre.
char giris[sizeof(sifre)]; //Şifre uzunluğu kadar dizi oluştur.//---Tuş takımı ayarları.---
const byte satir = 4; //Dört satır.
const byte sutun = 4; //Dört sütün.
char tusDizilimi[satir][sutun] = {//Tuş takımındaki sebollerin dizilimini tanımlama.
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
byte satirPinleri[satir] = {9, 8, 7, 6};
//{R1, R2, R3, R4} Satır pinlerinin Arduino pinlerine bağlantısı.
byte sutunPinleri[sutun] = {5, 4, 3, 2};
//{C1, C2, C3, C4} Sütün pinlerinin Arduino pinlerine bağlantısı.

//tusTakimi adında nesne oluşturma.
Keypad tusTakimi = Keypad( makeKeymap(tusDizilimi), satirPinleri, sutunPinleri, satir, sutun);
void setup() {
  Serial.begin(9600);
  Serial.println("Lütfen şifrenizi giriniz: ");
}
void loop() {
  char tus = tusTakimi.waitForKey(); //Tuşa basılana kadar bekle.
  Serial.print(tus); //Basılan tuşu ekrana yazdır.
  giris[hane] = tus; //Sırayla her tuşu giris dizisine yükle.
  hane++;

  if (hane == sizeof(sifre) - 1) { //Şifre uzunluğu kadar tuşa basıldığında...
  if (!strcmp(giris, sifre)) { //İki diziyi karşılaştır.
   Serial.println("\nŞifre doğru. Kapı açıldı");
   digitalWrite(LED, HIGH);
   delay(1000);
   Serial.println("Lütfen şifrenizi giriniz: ");
  } else Serial.println("\nŞifre yanlış. Lütfen tekrar deneyiniz.");
  hane = 0;
  }
}
```

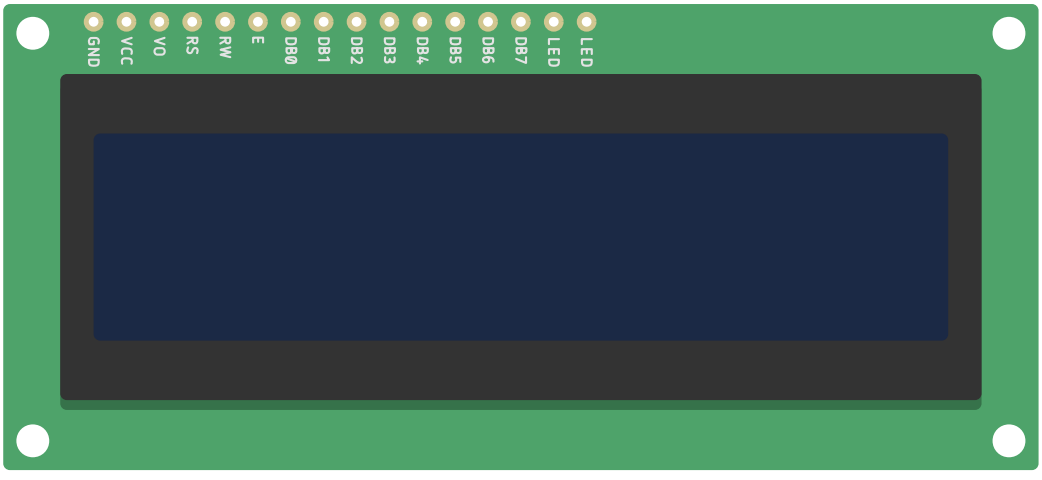
Sorular

1. Tuş takımında "1" rakamına basıldığında farklı bir karakter yazdırılabilir mi? Nedenini açıklayınız.
2. Kütüphane ile fonksiyon arasındaki fark nedir? Belirtiniz.

3.3. LCD EKRAN UYGULAMALARI

LCD [Liquid Crystal Display (likwıd kristıl displey) Sıvı Kristal Ekran], elektrikle kutuplanan sıvının ışığı tek fazlı geçirmesi ve önüne eklenen bir kutuplanma filtresiyle gözle görülebilmesi ilkesine dayanan bir görüntü teknolojisidir. Sıvı kristal ekranlar, düşük enerji tüketimine sahiptir. Görsel 3.10'daki LCD ekran, 2 satır ve 16 sütundan oluşmaktadır. Bir karakter ise 5x7 pikselden oluşmaktadır.

LCD modülde sekiz veri girişi bulunsa da dört girişli mod sıklıkla kullanılır. Dört girişli modda, veriler bir seferde yarım **bayt** olarak gönderilir. Bu nedenle dört veri bağlantısı yapılır. Veri girişinin üst yarısı (D4 - D7) kullanılırken diğer dört pin kullanılmamaktadır.



Görsel 3.10: LCD modül pinleri

1. GND: Topraklama pinidir. Bazı modüllerde VSS olarak etiketlenmiştir.

2. VCC: 5 V'luk güç bağlantısıdır. Bazı modüllerde VDD olarak etiketlenmiştir.

3. V0: Ekran parlaklığını kontrol etmek için 0 ile 5 V arasında değişen parlaklık kontrol voltajı girişidir.

4. RS: Register select pinidir. Giriş verilerinin LCD'de görüntülenip görüntülenemeyeceğini veya kontrol karakterleri olarak kullanılıp kullanılamayacağını kontrol eder.

5.RW: LCD'yi okuma veya yazma moduna geçirir. Genelde GND'ye bağlıdır. Okuma modundadır.

6. E: Etkinleştir pini. Lojik 1 olduğunda veri pinlerine uygulanan verileri okur. Lojik 0 olduğunda komutları yürütür veya verileri görüntüler.

7. D0: Veri girişi 0.

8. D1: Veri girişi 1.

9. D2: Veri girişi 2.

10. D3: Veri girişi 3.

11. D4: Veri girişi 4.

12. D5: Veri girişi 5.

13. D6: Veri girişi 6.

14. D7: Veri girişi 7.

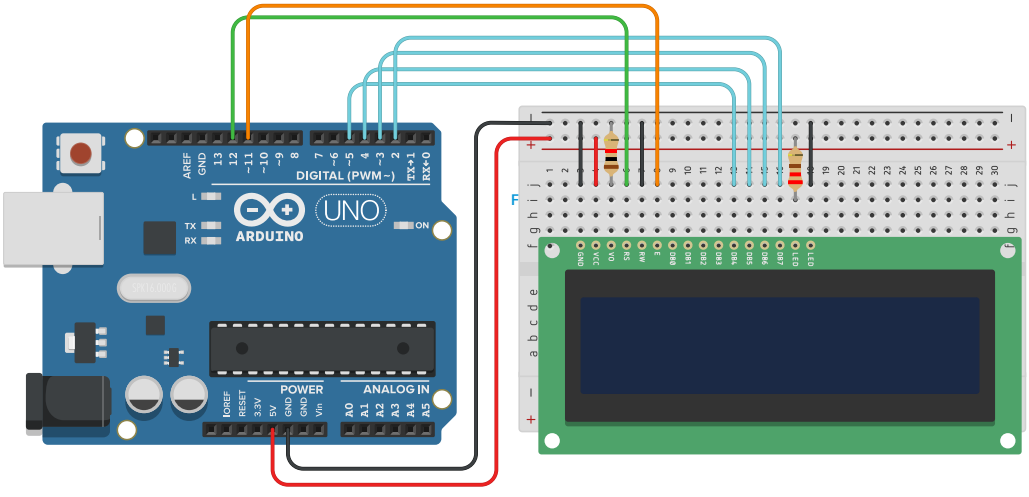
15. LED A: Arka ışık LED'ine anot (pozitif voltaj) bağlantısı.

16. LED K: Arka ışık LED'ine katot (GND) bağlantısı.

Uygulama Adı: LCD Uygulaması — No.: 2

Amaç: LCD uygulaması yapmak.

Görsel 3.11'deki LCD uygulamasında V0 ucuna bağlı dirençle ekran parlaklığı ayarlanmıştır. Bu pine potansiyometre bağlanarak da LCD ekranın parlaklığı ayarlanabilir. LCD kütüphanesi dâhilî kütüphanelerde mevcuttur. Menülerden "Taslak → Library Ekle → LiquidCrystal" seçilerek kütüphane taslakta kullanılır. Aşağıdaki program LCD ekrana "Merhaba dünya." yazdırmaktadır. Bu programın bir örneğine menülerden "Dosya → Örnekler → LiquidCrystal → HelloWorld" adımlarını izleyerek de ulaşılır.



Görsel 3.11: LCD uygulaması

LCD uygulaması programı aşağıdaki gibidir:

```
#include <LiquidCrystal.h> //v1.0.7
// LCD modülün Arduino pinlerine bağlantısı.
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup() {
  lcd.begin(16, 2); //LCD modülün satır ve sütun sayısı.
  lcd.print("Merhaba Dunya!"); //LCD ekrana yazdırılan metin.
}
void loop() {
  lcd.setCursor(0, 1);
  //imleci 0. sütun 1. satıra (0. satır üstteki, 1. satır alttakidir.) konumlandır.
  lcd.print(millis() / 1000); //Geçen süreyi saniye olarak yaz.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.11'deki devreyi kurunuz. Programı yükleyiniz.
4. Ekrana adınızı yazdırınız.

SIRA SİZDE

1. Ekrana kayan yazı yazdıran uygulamayı gerçekleştiriniz. Programa "Dosya → Örnekler → LiquidCrystal → Scroll" adımlarını izleyerek ulaşabilirsiniz.

```
#include <LiquidCrystal.h>
// LCD modülün Arduino pinleme bağlantısı.
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup() {
  lcd.begin(16, 2); //LCD modülün satır ve sütun sayısı.
  lcd.print("Merhaba Dunya!"); //LCD ekrana yazdırılan metin.
  delay(1000);
}
void loop() {
  // 15 hücre sola kaydır. (Metin uzunluğu kadar.)
  for (int i = 0; i < 15; i++) {
  lcd.scrollDisplayLeft();
  delay(150);
  }
  // 31 hücre sağa kaydır. (Metin uzunluğu + ekran uzunluğu kadar.)
  for (int i = 0; i < 31; i++) {
  lcd.scrollDisplayRight();
  delay(150);
  }
  for (int i = 0; i < 16; i++) { // 16 hücre sola kaydır.(Ekran uzunluğu kadar.)
  lcd.scrollDisplayLeft();
  delay(150);
  }
  delay(1000);
}
```

2. Ekrana özel karakter yazdıran uygulamayı gerçekleştiriniz. Programa "Dosya → Örnekler → LiquidCrystal → CustomCharacter" adımlarını izleyerek ulaşabilirsiniz.

```
#include <LiquidCrystal.h>
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

/* Bir taslakta en fazla sekiz karakter oluşturulabilir. Bir karakter 5 x 8 pikseldir.*/
byte kalp[8] = {
  0b00000,
  0b01010,
  0b11111,
  0b11111,
  0b11111,
  0b01110,
```

```
  0b00100,
  0b00000
};
byte gulenYuz[8] = {
  0b00000,
  0b00000,
  0b01010,
  0b00000,
  0b00000,
  0b10001,
  0b01110,
  0b00000
};
byte kollarAsagi[8] = {
  0b00100,
  0b01010,
  0b00100,
  0b00100,
  0b01110,
  0b10101,
  0b00100,
  0b01010
};
byte kollarYukari[8] = {
  0b00100,
  0b01010,
  0b00100,
  0b10101,
  0b01110,
  0b00100,
  0b00100,
  0b01010
};
void setup() {
  lcd.begin(16, 2);
  lcd.createChar(0, kalp);
  lcd.createChar(1, gulenYuz);
  lcd.createChar(2, kollarAsagi);
  lcd.createChar(3, kollarYukari);
  lcd.setCursor(0, 0); // İmleci sol üste konumlandır.
  lcd.print("I ");
  lcd.write(byte(0)); // 0. karakter byte olarak çağırılır.
  lcd.print(" Arduino! ");
  lcd.write(1);
}
```

```
void loop() {
  int pot = analogRead(A0); // Pot bağlanabilir, boş da bırakılabilir.
  int sure = map(pot, 0, 1023, 200, 1000); // Pot'la animasyon hızını ayarla.
  lcd.setCursor(7, 1); // İmleci alt ortaya konumlandır.
  lcd.write(2); // Kollar aşağı.
  delay(sure);
  lcd.setCursor(7, 1);
  lcd.write(3); // Kollar yukarı.
  delay(sure);
}
```

3. Aşağıdaki kod nasıl çalışmaktadır?

```
#include <LiquidCrystal_I2C.h> // v1.1.2

LiquidCrystal_I2C lcd(0x3F, 16, 2); // I2C adresi 0x3F. 16x2 LCD ekran.

void setup() {
  lcd.init();
  lcd.clear();
  lcd.backlight();

  lcd.setCursor(2, 0);  // 0. satır 2. karakter.
  lcd.print("Merhaba Dünya!");

  lcd.setCursor(2, 1);  // 1. satır 2. karakter.
  lcd.print("I2C modül.");
}

void loop() {
}
```

Sorular

1. LCD ekrana I2C modülü kullanılarak nasıl yazı yazdırılır? Araştırınız.

2. Aşağıdaki kod nasıl çalışmaktadır?

```
#include <LiquidCrystal_I2C.h> // v1.1.2
LiquidCrystal_I2C lcd(0x3F, 16, 2); // I2C adresi 0x3F. 16x2 LCD ekran.
void setup() {
  lcd.init();
  lcd.clear();
  lcd.backlight();
  lcd.setCursor(2, 0);  // 0. satır 2. karakter.
  lcd.print("Merhaba Dünya!");
  lcd.setCursor(2, 1);  // 1. satır 2. karakter.
  lcd.print("I2C modül.");
}
void loop() {}
```

3.4. SENSÖR UYGULAMALARI

LCD sensörler; etraftaki ısı, ışık, nem, ses, basınç, kuvvet, elektrik, uzaklık vb. fiziksel büyüklüklerin değişimini algılar. Sensörlerden elde edilen veriler elektronik bir modülle elektrik sinyaline dönüştürüldükten sonra mikrodenetleyiciye giriş bilgisi olarak verilir. Aktif sensörler, GND ve VCC bağlantı uçları olan ve çıkışta analog veya dijital uçları bulunan modüler yapıdaki sensörlerdir. Pasif sensörler ise iki ucuna uygulanan potansiyel farkı sensörün algıladığı fiziksel büyüklüğün durumuna göre sensörün iki ucu arasında minimum ile maksimum arası akım geçirecek şekilde çalışır (Buton, anahtar, potansiyometre, NTC/PTC, basınç sensörleri, LDR, fototransistör, fotodiyot, mikrofon vb.). Tablo 3.1'de sensör çeşitleri verilmiştir.

Tablo 3.1: Algıladığı Fiziksel Büyüklüklere Göre Sensör ve Transdüser Çeşitleri

Fiziksel Değişken	Sensör, Transdüser
Sıcaklık	Termokupl (ısıl çift) Termistor (NTC, PTC) Termostat RTD [resistive temperature detectors (resistiv temperıtur dıtektırs)]
Ses	Dinamik mikrofon Kapasitif mikrofon Hoparlör Piezo kristal
Işık (Optik)	LDR Foto diyot Foto transistör Fotovoltaik pil (solar cell)
Basınç (Gerilme)	Gerilme ölçer [strain gauge (sitreyn geyç)] Basınç anahtarı (pressure switch) Yük hücresi (loadcells)
Manyetizma	Reed röle Hall Sensörü
Yaklaşım	Endüktif sensörler Kapasitif sensörler Optik sensörler

Uygulama Adı: Isı Sensörü Uygulaması No.: 3

Amaç: Isı sensörü uygulaması yapmak.

Görsel 3.12'de LM35 sıcaklık sensörü pin yapısı, Görsel 3.13'te ise uygulama devresi görülmektedir. LM35 sıcaklık sensörü 4-20 V'la beslenebilmektedir. Çıkış ucu ise 25 °C'de 250 mV gerilim vermektedir. Bu değer her 1 °C'de 10 mV değişim gösterir.

10 bitlik analog giriş 0-5 V için 0-1023 arası değer almaktadır. Analog girişteki gerilim (5/ 1023) *A0 işlemiyle volt cinsinden bulunur. Volt değeri 1000'le çarpıldığında mV değeri elde edilir. LM35'te her 10 mV'ta sıcaklık değeri 1 °C değiştiğinden A0 girişindeki mV değeri 10'a bölündüğünde sıcaklık değeri elde edilir. Tek formüle indirgenirse Sıcaklık = (5/1024) *A0 formülü elde edilir.

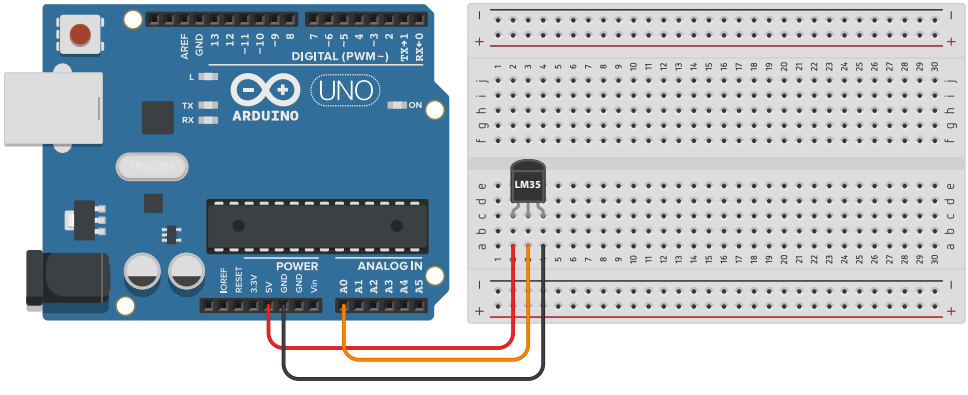


Görsel 3.12: LM35 pin yapısı

+V: 5 V besleme.

OUT: Analog veri çıkışı.

GND: Topraklama pini.



Görsel 3.13: LM35 uygulaması

Isı sensörü uygulaması programı aşağıdaki gibidir:

```
void setup() {
  Serial.begin(9600);
}
```

```
void loop() {
  int deger = analogRead(A0); // Analog girişi oku.
  float gerilim = (5000.0 / 1023.0) * deger; // Analog girişteki gerilim (mV).
  float  sicaklik = gerilim / 10.0; // Sıcaklık °C
  Serial.print(sicaklik);
  Serial.println(“ °C”);
  delay(1000);
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.13'teki devreyi kurunuz. Programı yükleyiniz.
4. LM35'i ısıtarak sıcaklık değişimini seri port ekranından gözlemleyiniz.
5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Sıcaklığı LCD ekrana yazdıran uygulamayı gerçekleştiriniz.

```
#include <LiquidCrystal.h>
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
  lcd.begin(16, 2); //LCD modülün satır ve sütün sayısı.
}
void loop() {
  lcd.setCursor(0, 0);
  lcd.print((500.0 / 1023.0) * A0); //Sıcaklık değerini yaz.
  lcd.setCursor(4, 0);
  lcd.print(“°C”);
}
```

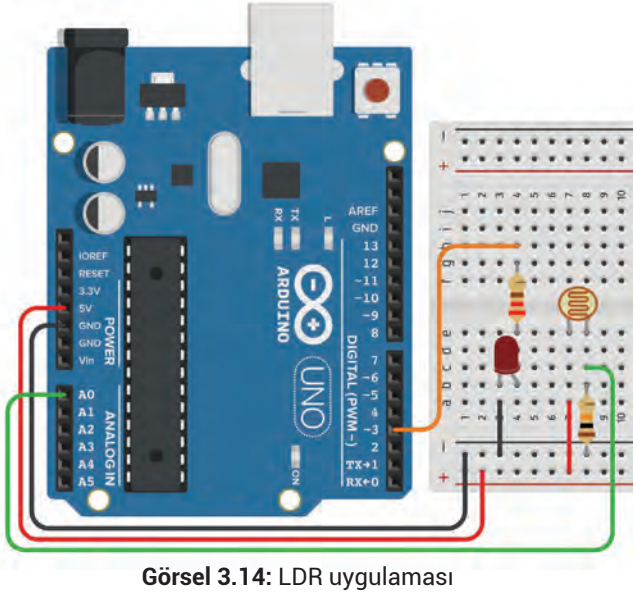
Soru

1. LM35 sıcaklık sensöründe sıcaklık - direnç ilişkisi nasıldır? Belirtiniz.

Uygulama Adı: Işık Sensörü Uygulaması No.: 4

Amaç: Işık sensörü uygulaması yapmak.

Görsel 3.14'teki devrede Arduino'nun 5 V gerilimi üzerine uygulanan LDR'nin direnç-gerilim değişiminin anlaşılabilmesi için 10 kΩ'luk bir direnç seri bağlanarak gerilim bölücü oluşturulur. Bağlantı noktasının gerilim değişimi Arduino'nun analog girişine verilir. Aydınlık ortamda LDR'nin direnci 10 kΩ'un altındadır. Karanlıkta (LDR elle kapatıldığında) LDR direnci MΩ değerine doğru artar. LDR üzerine düşen gerilim artacağı için LDR ucuna bağlı A0 girişinden alınan bu bilgi 10 bitlik ADC'le 1023 değerine yaklaşır. Gelen bilginin ¼'ü 3 numaralı PWM çıkışından LED'e verilir. Böylelikte karanlık arttıkça LED'in parlaklığı da artar.



Görsel 3.14: LDR uygulaması

LDR uygulaması programı aşağıdaki gibidir:

```
const byte LED = 3;
void setup() {
  pinMode(LED, OUTPUT);
}
void loop() {
  // Analog giriş bilgisini 1/4'ü LDR değişkenine yükle.
  byte LDR = analogRead(A0) / 4;
  analogWrite(LED, LDR);
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.14'teki devreyi kurunuz. Programı yükleyiniz.

4. Elinizi LDR'ye yaklaştırıp uzaklaştırarak LED'in durumunu gözlemleyiniz.

5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Akşam olduğunda lambayı yakan uygulamanın programını yazınız.

```
const byte lambaPini = 13;

void setup() {
  Serial.begin(9600); //Seri iletişim başlatıldı.
  pinMode(lambaPini, OUTPUT);
}

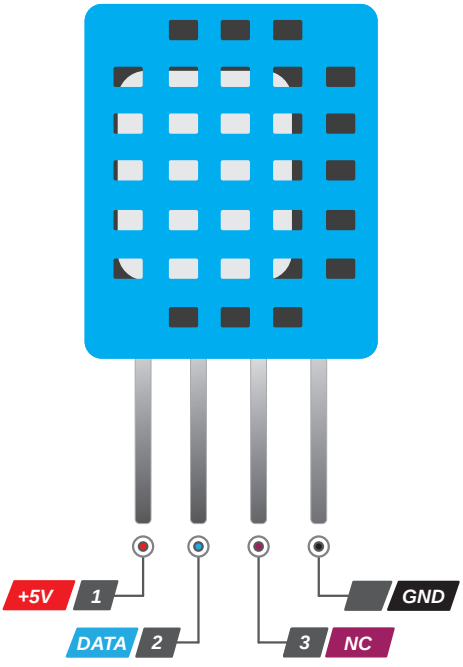
void loop() {
  int LDR = analogRead(A0); //A0 girişinden okunan 0-1023 arası veriyi LDR değişkenine yükle.
  Serial.println(LDR); //LDR değişkeninin içeriğini yazdır.
  if (LDR < 500) { // 500'den küçükse...
    digitalWrite(lambaPini, HIGH); // Lambayı yak.
  }
  else { // Lambayı söndür.
    digitalWrite(lambaPini, LOW);
  }
}
```

Soru

1. PWM çıkışına analogWrite(LED, 382); fonksiyonuyla 382 sayısı gönderilirse LED'e kaç volt verilmiş olur? Hesaplayınız.

Nem Sensörü Uygulaması

DHT sensörleri, kapasitif nem sensörü ve termistör (NTC) olmak üzere iki parçadan oluşur. İçinde analog dijital dönüştürü vardır. Görsel 3.15a'da DHT11 ve DHT22'nin pin yapısı görülmektedir. DHT11 ve DHT22 aynı pin çıkışına sahiptir ancak bunların farklı özellikleri vardır. DHT11 0 ile 50 °C arasındaki sıcaklığı ±2 °C doğrulukla verirken DHT22 -40 ile 80 °C arasındaki sıcaklığı ±0,5 °C doğrulukla verir. DHT11 saniyede bir örnekleme yaparken DHT22 saniyede iki örnekleme yapar.



Görsel 3.15a: DHT11 ve DHT22'nin pin yapısı

VCC: 5 V besleme.

Data: Dijital veri çıkışı.

NC: Kullanılmayan pin.

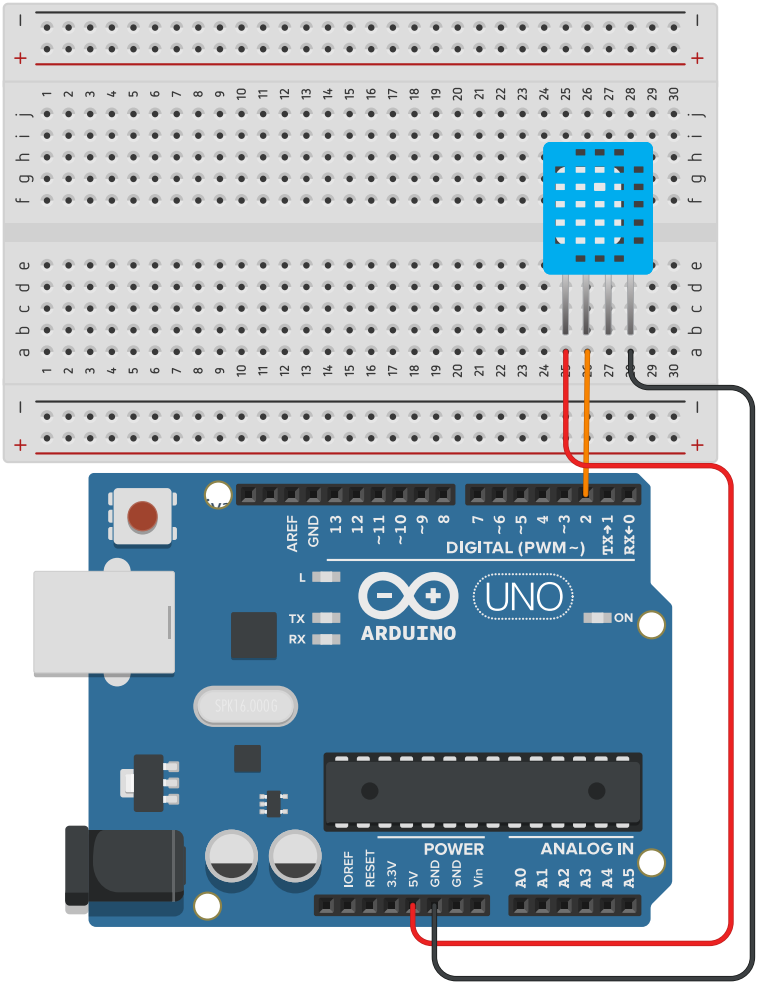
GND: Topraklama pini.

Menülerden "Taslak→Library Ekle → Kütüphaneleri Yönet" seçeneğinden arama kutusuna "DHT" yazılıp "Adafruit DHT Sensör Library" kütüphanesi bulunarak versiyon 1.2.3 kurulur (Kütüphanenin son versiyonunda aşağıdaki kod çalışmaz.). Daha sonra "Taslak → Library Ekle → DHT Sensör Library" seçilerek programın başına eklenir. Bu programın bir örneğine menülerden "Dosya →Örnekler →DHT Sensör Library→DHT Tester" adımlarını izleyerek de ulaşılır.

Uygulama Adı: Nem Sensörü Uygulaması No.: 5

Amaç: Nem sensörü uygulaması yapmak.

Görsel 3.15b'deki devre uygulama ortamının sıcaklık ve nem bilgisini verir.



Görsel 3.15b: DHT11 ve DHT22 uygulaması

Nem sensörü uygulaması programı aşağıdaki gibidir:

```
#include "DHT.h" // Adafruit DHT Sensor Library v1.2.3
#define DHTpin 2     // Sensörün bağlandığı pin.
#define DHTtipi DHT11   // Sensör tipi: DHT11, DHT21, DHT22.

DHT dht(DHTpin, DHTtipi); //dht isimli nesne oluşturuldu.
void setup() {
  Serial.begin(9600);
  dht.begin();
```

```
 }
void loop() {
  delay(1000);
 int nem = dht.readHumidity(); // Nem bilgisini oku.
  int sicaklik = dht.readTemperature(); // Sıcaklık bilgisini oku.

  if (!(sicaklik || nem)) { // Sıcaklık ve nem bilgisi yoksa...
  Serial.println("DHT sensor okunamadı!");
  return; // Başa dön.
  }

  Serial.print("Sıcaklık: ");
  Serial.print(sicaklik);
  Serial.print(" °C \t");
  Serial.print("Nem: %");
  Serial.println(nem);
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.15b'deki devreyi kurunuz. Programı yükleyiniz.
4. Ortamın nem ve sıcaklık bilgisini seri port ekranından gözlemleyiniz.
5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Sıcaklık ve nem bilgisini LCD ekrana yazdıran uygulamayı gerçekleştiriniz.

Soru

1. Nem sensörünün içinde hangi tip termistör kullanılmaktadır? Belirtiniz.
2. DHT11 ve DHT22 arasındaki farklar nelerdir? Belirtiniz.

Hareket Sensörü

Maddeler sahip oldukları ısıdan dolayı insanların görebileceği ışık aralığının altında kızılötesi ışık yayar. Normal sıcaklığındaki insan vücudu (36,5 °C) 10 mikrometre dalga boyunda ışıma yapar. Hareket sensörü bulunduğu ortamdaki kızılötesi ışık dalgalarını içindeki özel kristal malzemeyle (piroelektrik sensör) elektriğe dönüştürür. Ortamdaki kızılötesi ışık miktarı sabitken bir canlı ortama girdiğinde kızılötesi ışık miktarında artış olur. Hareket sensörü bu artışı algılayarak çıkış verir. Hareket sensörünün diğer adı PIR (pasif infrared sensor-pasif kızılötesi) sensördür. PIR, kızılötesi ışık gönderip yansımasını almadığı için pasif moddadır yani sadece ortamdaki kızılötesi ışık değişimini algılar.

Görsel 3.16'da yaygın olarak kullanılan HC-SR501 hareket sensörünün pin yapısı ve ayarları görülmektedir. HC-SR501 sensörü temel olarak **piroelektrik sensör** ve kızılötesi sinyalleri piroelektrik sensöre odaklayan **fresnel lens** (sensörün dışındaki plastik kısım) olmak üzere iki ana bölümden oluşur.Kartta iki potansiyometre vardır. Biri hareketin algılanabileceği en fazla mesafeyi ayarlar. Bu mesafe 3 metre ile yaklaşık 7 metre arasında değişmektedir. Diğer potansiyometre ise hareket algılandıktan sonra çıktının ne kadar süreyle lojik 1'de kalacağını belirler. Bu süre de en az 3 saniye, en fazla 5 dakikadır.

Hareket sensörünün arkasında bulunan **jumper**la iki ayardan biri seçilir.

H: Bu konumda HC-SR501, hareketi algılamaya devam ettiği sürece lojik 1 sinyal vermeye devam eder.

L: Bu konumda çıkış, zaman potansiyometresi ayarıyla ayarlanan süre boyunca lojik 1'de kalacaktır.

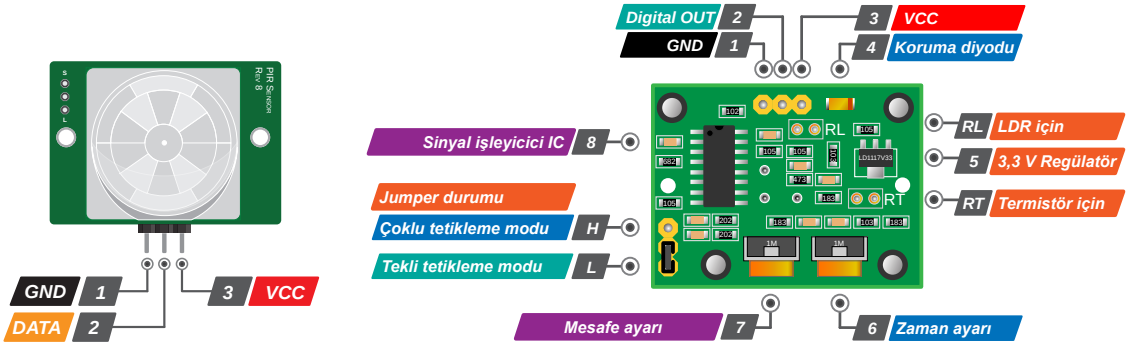
Hareket sensörünün çıkışındaki lojik 1 seviyesi 3,3 V'tur. 3,3 V gerilim, giriş olarak ayarlanan Arduino pininde lojik 1 seviyesi olarak algılanır çünkü Arduino'da dijital giriş olarak ayarlanmış bir pinde 3 V üzerindeki gerilimler lojik 1 olarak algılanır.

HC-SR501'e güç verildiğinde ortamdaki kızılötesi enerjiye alışması 30 ila 60 saniye sürer. Ayrıca sensörün bir okuma yaptıktan sonra yaklaşık 5 veya 6 saniye süren "sıfırlama" süresi vardır. Bu süre zarfında sensör herhangi bir hareket algılamayacaktır.

HC-SR501 devre kartında iki ek bileşen için lehim yuvaları bulunur.

RT: Sıcaklığa duyarlı direnç (termistör) içindir. Termistör eklemek, HC-SR501'in istenen sıcaklıklarda kullanılmasına izin verir ve dedektörün doğruluğunu bir dereceye kadar artırır.

RL: Işığa bağımlı direnç (LDR) içindir. LDR eklendiğinde HC-SR501 yalnızca karanlıkta çalışır.



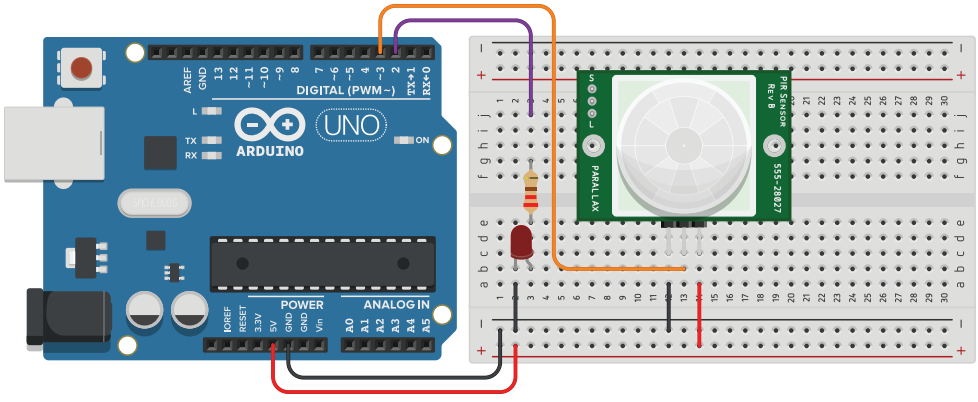
Görsel 3.16: Hareket sensörü pin yapısı ve ayarları

VCC: 5 V - 12 V besleme. **DATA:** Dijital veri çıkışı. (3,3 V) **GND:** Topraklama pini.

Uygulama Adı: Hareket Sensörü Uygulaması No.: 6

Amaç: Hareket sensörü uygulaması yapmak.

Görsel 3.17'deki devrede hareket sensörü, ortamda bir hareket algıladığında LED'i yakmakta ve seri port ekranından bilgi vermektedir.



Görsel 3.17: Hareket sensörü uygulaması

Hareket sensörü uygulaması programı aşağıdaki gibidir:

```
const byte LED = 2, PIR = 3;
void setup() {
  pinMode(LED, OUTPUT);
  pinMode(PIR, INPUT);
  Serial.begin(9600);
}
void loop() {
  if (digitalRead(PIR))  { // PIR çıkış veriyorsa...
  digitalWrite(LED, HIGH);
  Serial.println("Hareket algılandı.");
  }
  else  { // PIR çıkış vermiyorsa...
  digitalWrite(LED, LOW);
  Serial.println("Hareket algılanmadı.");
  }
```

İşlem Basamakları

1. Görsel 3.17'deki devreyi kurunuz. Programı yükleyiniz.
2. Hareketsiz bir ortamda HC-SR501 sensörünün önüne geçerek çalışmasını gözlemleyiniz.

SIRA SİZDE

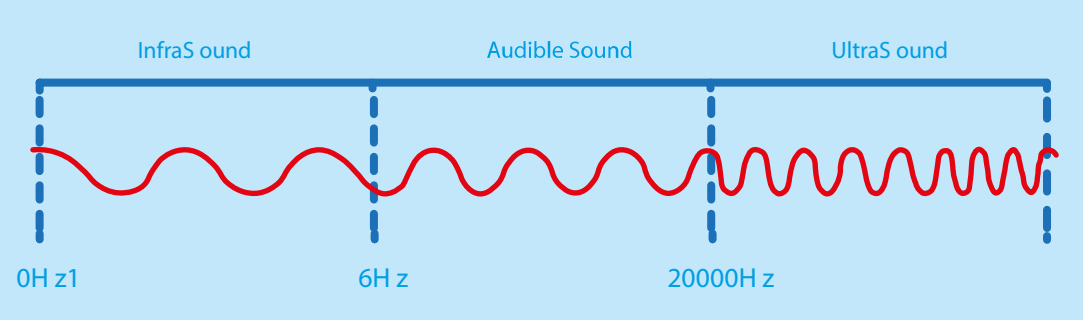
1. Uygulamayı hareket algılandığında sesle uyarı verecek şekilde düzenleyiniz.

Soru

1. Maddeler hangi sıcaklıkta kızılötesi ışınım yaymaz? Araştırınız.

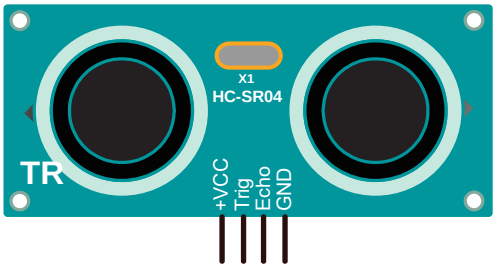
Ultrasonik Sensör

İnsanlar, frekansı 16 Hz ile 20 kHz arasında olan sesleri duyabilmektedir. Ultrasonik ses, insanın işitme sınırından daha yüksek frekanslardaki ses dalgalarıdır. Görsel 3.18'de ses frekans aralıkları görülmektedir.



Görsel 3.18: Ses frekans aralıkları

Görsel 3.19'da HC-SR04 ultrasonik sensör ve pin yapısı görülmektedir. HC-SR04 ultrasonik sensör üzerinde alıcı ve verici olmak üzere iki yüzey bulunur. Verici yüzeyden 15 derece açıyla ortama 40 kHz frekansta ultrasonik ses dalgası salınır. Alıcı, iletilen darbeleri dinler. Yayılan ses dalgası bu alanda bulunan bir cisme çarptığında ses dalgası cismin yüzeyinden sensöre geri yansır. Yansıyan dalganın giriş yüzeyine gelene kadar katettiği mesafenin belirlenebilmesi için genişliği kullanılabilen bir çıkış darbesi üretilir. Üretilen bu dalganın süresi ölçülerek hız formülü yardımıyla cismin uzaklığı hesaplanır. Sensör, 2 cm ile 400 cm arasındaki uzaklıkları 3 mm hassasiyetle ölçebilmektedir.



Görsel 3.19: HC-SR04 ultrasonik sensör ve pin yapısı

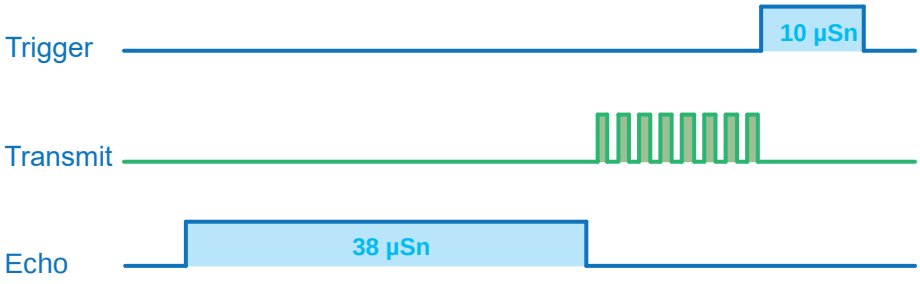
VCC: 5 V besleme.

Trig: Ultrasonik ses darbelerini tetiklemek için kullanılır.

Echo: Yansıyan sinyal alındığında darbe üreten pindir. Darbenin uzunluğu, iletilen sinyalin algılanması için geçen süreyle orantılıdır.

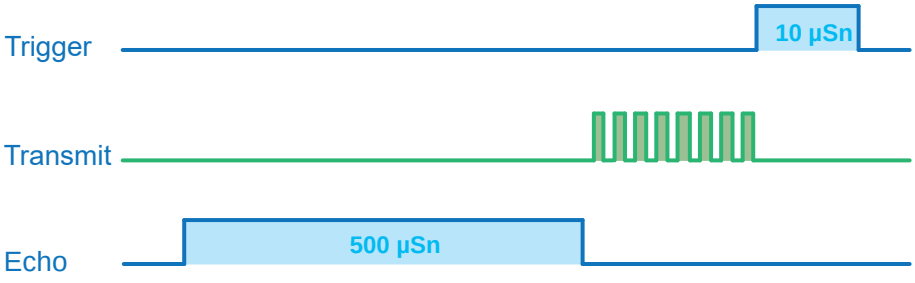
GND: Topraklama pini.

Trig pinine en az 10 µS (10 mikrosaniye) süreli bir darbe uygulandığında sensör, 40 kHz'de sekiz darbelik bir ses kalıbını ortama gönderir. Bu sekiz darbeli model, alıcının iletilen modeli ortamdaki ultrasonik gürültüden ayırt etmesini sağlar. Bu sinyal gönderildikten sonra **Echo** pini lojik 1 konumuna geçer ve gönderilen sinyalin dönmesini bekler. Eğer ses 38 milisaniye (sensörün menzil süresi) içinde geri dönmezse engel yoktur ve işlem tekrar eder (Görsel 3.20).



Görsel 3.20: Ultrasonik sensör engel algılamadığında

Darbeler geri yansıtılırsa sinyal alınır alınmaz Echo pini lojik 0 olur. Geçen süreye bağlı olarak **Echo** pini tarafından genişliği 150 µs ile 25 ms arasında değişen bir darbe üretilir (Görsel 3.21).



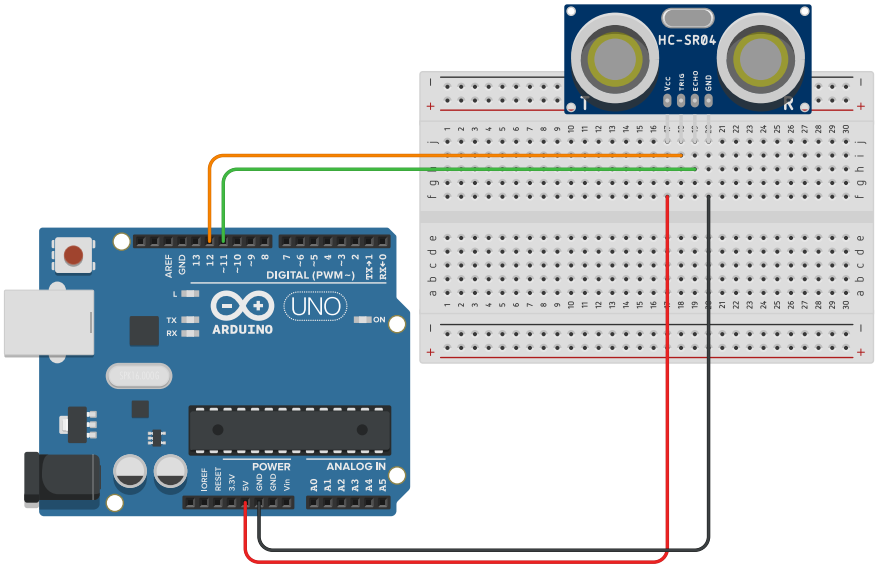
Görsel 3.21: Ultrasonik sensör engel algıladığında

Echo pinin lojik 1 seviyesinde kalma süresi ses dalgasının engele çarpıp geri dönme süresidir. **"Yol= hız x zaman"** formülünde bu süre kullanılarak mesafe hesaplanır. Sesin hızı ortamdaki nem ve sıcaklığa göre değişmektedir. 20 °C'de sesin yayılma hızı **344 m/s'dir**. DHT22 gibi bir sensör ilave edilerek ve ses hızı m/s = 331.4 + (0.606 * sıcaklık) + (0.0124 * nem) formülü programda kullanılarak ölçüm doğruluğu artırılabilir. Mesafe, süre, ortalama süre vb. değerleri veren NewPing.h kütüphanesi HC-SR04 uygulamalarında kullanılmaktadır.

Uygulama Adı: Ultrasonik Sensör Uygulaması No.: 7

Amaç: Ultrasonik sensör uygulaması yapmak.

Görsel 3.22'deki uygulamada kullanılan NewPing kütüphanesini kurmak için menülerden "Taslak → Library Ekle → Kütüphaneleri Yönet" seçeneğinden arama kutusuna "NewPing" yazılır. "NewPing" kütüphanesi bulunarak kurulur. Daha sonra "Taslak → Library Ekle → NewPing" seçilerek programın başına eklenir. Bu programın bir örneğine menülerden "Dosya → Örnekler → NewPing → NewPingExample" adımları izlenerek de ulaşılır. NewPing kütüphanesi, **trig** ve **echo** pinlerini aynı pin numarasında sırayla kullanarak birden fazla sensör kullanımında pin sayısından tasarruf eder.



Görsel 3.22: Ultrasonik sensör uygulaması

Ultrasonik sensör uygulaması programı aşağıdaki gibidir:

```
#include <NewPing.h> // v1.9.1
const byte trig = 12, echo = 11, sinir = 200;
NewPing olcum(trig, echo, sinir); // olcum isimli nesne oluşturuldu.
void setup() {
  Serial.begin(9600);
}
void loop() {
  int uzaklik = olcum.ping_cm(); // cm cinsinden uzaklık.
  Serial.print(uzaklik);
  Serial.println(" cm");
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.22'deki devreyi kurunuz. Programı yükleyiniz.

4. Ultrasonik sensörün önüne kitap, defter vb. engel koyunuz.
5. Engeli ileri geri hareket ettirip mesafeyi seri ekrandan gözlemleyiniz.
6. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Ultrasonik sensör uygulamasını kütüphane kullanmadan gerçekleştiriniz.

```
const byte trig = 12, echo = 11;
unsigned long sure;
int uzaklik;
void setup() {
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  Serial.begin(9600);
}
void loop() {
  digitalWrite(trig, 0); // Darbe sinyali gönder.
  delayMicroseconds(10);
  digitalWrite(trig, 1);
  delayMicroseconds(10);
  sure = pulseInLong(echo, 1); // Gelen sinyalin HIGH'da kalma süresini oku.
  uzaklik = (0.0343 * sure) / 2; // Mesafeyi hesapla.
  Serial.print(uzaklik);
  Serial.println(" cm");
```

2. 2 numaralı pine **buzzer** ekleyerek uygulamayı park sensörüne dönüştürünüz.

```
#include <NewPing.h>
const byte buzzer = 2, trig = 12, echo = 11, sinir = 200;
NewPing olcum(trig, echo, sinir); // olcum isimli nesne oluşturuldu.
void setup() {
  pinMode(buzzer, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  byte uzaklik = olcum.ping_cm(); // cm cinsinden uzaklık. (En fazla 200.)
  Serial.print(uzaklik);
  Serial.println(" cm");
  digitalWrite(buzzer, 1);
  delay(5 * uzaklik); // Mesafe azaldıkça bip sesini hızlandır.
  digitalWrite(buzzer, 0);
  delay(5 * uzaklik);
}
```

3. 3 adet ultrasonik sensörün mesafe bilgisini ekrana yazdırınız.

```
#include <NewPing.h>
byte sinir = 200;
NewPing sensor[3] = {   // Üç adet sensor nesnesi oluşturuldu.
  NewPing(2, 2, sinir), // Trig ve echo pinleri aynı pine bağlı.
  NewPing(3, 3, sinir),
  NewPing(4, 4, sinir)
};
void setup() {
  Serial.begin(115200); // Seri ekran  115200 baudrate'e ayarlandı.
}
void loop() {
  for (byte i = 0; i < 3; i++) { // Üç sensörün bilgileri göster.
  delay(50); // En az 29 ms beklenmeli.
  Serial.print(i);
  Serial.print("=");
  Serial.print(sensor[i].ping_cm());
  Serial.print("cm ");
  }
  Serial.println();
```

4. Mesafeyi LCD ekrana yazdıran uygulamayı gerçekleştiriniz.

```
#include <LiquidCrystal.h>
const byte trigEchoTekPin = 10; // Trig ve Echo pinleri 10 numaralı pine bağlı.
unsigned long sure;

// LCD modülün Arduino pinlerne bağlantısı.
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup() {
  lcd.begin(16, 2); // LCD modülün satır ve sütun sayısı.
  lcd.print("Mesafe");
}
void loop() {
  pinMode(trigEchoTekPin, OUTPUT); // 10 numaralı pini çıkış yap.
  digitalWrite(trigEchoTekPin, 0); // Darbe sinyali gönder.
  delayMicroseconds(10);
  digitalWrite(trigEchoTekPin, 1);
  delayMicroseconds(10);
  pinMode(trigEchoTekPin, INPUT); // 10 numaralı pini giriş yap.

  sure = pulseInLong(trigEchoTekPin, 1); // Gelen sinyalin HIGH'da kalma süresini oku.
  lcd.setCursor(0, 1);
  lcd.print((0.0343 * sure) / 2); // Mesafeyi yazdır.
  lcd.setCursor(4, 0);
  lcd.print("cm");
}
```

5. Trig ve echo pinlerini tek bir pinde kullanınız.

```
const byte trig = 12, echo = 12;
unsigned long sure;
int uzaklik;

void setup() {
  Serial.begin(9600);
}

void loop() {
  pinMode(trig, OUTPUT); // 12 numaralı pini çıkış yap.
  digitalWrite(trig, 0); // Darbe sinyali gönder.
  delayMicroseconds(10);
  digitalWrite(trig, 1);
  delayMicroseconds(10);
  pinMode(echo, INPUT); // 12 numaralı pini giriş yap.

  sure = pulseInLong(echo, 1); // Gelen sinyalin HIGH'da kalma süresini oku.
  uzaklik = (0.0343 * sure) / 2; // Mesafeyi hesapla.

  Serial.print(uzaklik);
  Serial.println(" cm");
}
```

Sorular

1. Kütüphane kullanılarak yazılmış bir program, kütüphane kullanılmadan da yazılabilir mi? Açıklayınız.
2. Program yazılırken kütüphane kullanımı neden tercih edilir? Belirtiniz.
3. NewPing kütüphanesi engel algılamadığında (engel >400 cm) hangi değeri döndürmektedir? Belirtiniz.

3.5. ELEKTRİK MOTOR UYGULAMALARI

Elektrik enerjisini hareket enerjine çevirmeye yarayan mekanik aygıtlara motor denir. Elektrik motorları AC ve DC olarak iki gerilim türünde çalışacak şekilde üretilir. Diğer birçok motor modeli bu iki motor türünden geliştirilmiştir. AC motorlar genel olarak asenkron motor olarak iki türde üretilir. Bunlar bir fazlı ve üç fazlı motorlar olmak üzere iki türdedir. Bu motorların dönüş yönünü değiştirmek için farklı uygulamalar mevcuttur.

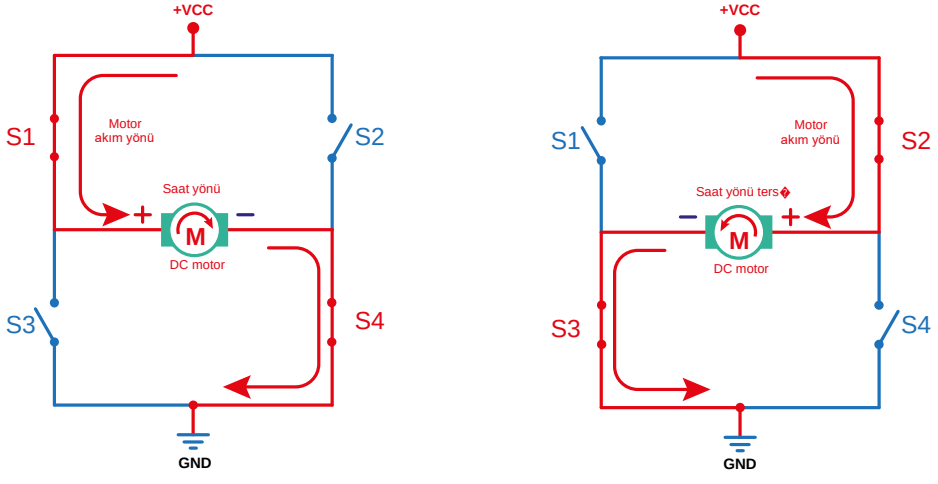
DC motorlar ise senkron motorlar, step motorlar, fırçalı ve fırçasız motorlar, sabit mıknatıslı motorlar olmak üzere üretilmektedir. Fırçalı DC motorlarda motor yönünü değiştirmek için besleme gerilim uçlarının yerini değiştirmek yeterlidir. Step motorlar, fırçasız motorlar ve sabit mıknatıslı motorlarda ise motor yönünü değiştirmek için farklı motor sürme teknikleri uygulanmaktadır.

Bu uygulamada fırçalı DC motor dönüş yönünü değiştirmek için dört transistörlü H tipi köprü devresi kullanılmaktadır.

DC Motor Uygulaması

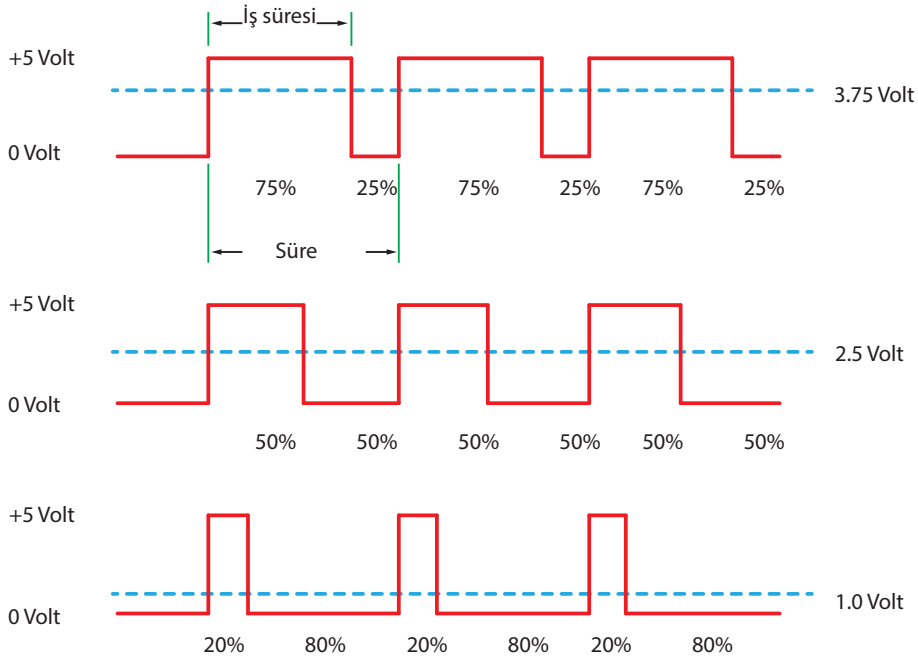
DC motor, doğru akım elektrik enerjisini mekanik enerjiye dönüştüren elektrik motorudur. Motorun içinde yer alan sargılara elektrik akımı uygulandığında manyetik kuvvetin etkisiyle motor hareket eder. DC motorda + ve - kutupların yerleri değiştirilerek dönüş yönü değiştirilirken uygulanan gerilim değiştirilerek de motorun hızı kontrol edilir.

DC motorun dönüş yönünü değiştirmek için H-köprüsü (H-Bridge) kullanılır. H köprüsü devresi, merkezinde motor bulunan ve "H" harfi benzeri bir düzenleme oluşturan dört anahtar içerir. Görsel 3.23'te çapraz iki anahtarın aynı anda kapatılması, motora uygulanan voltajın polaritesini tersine çevirir ve motorun dönüş yönünün değişmesini sağlar.



Görsel 3.23: DC motor yönü değiştirme (H-köprüsü)

Görsel 3.24'te motorun hızını ayarlamak için motora uygulanan gerilim değişiminin mikrodenetleyicilerde PWM kullanılarak yapılışı gösterilmiştir.

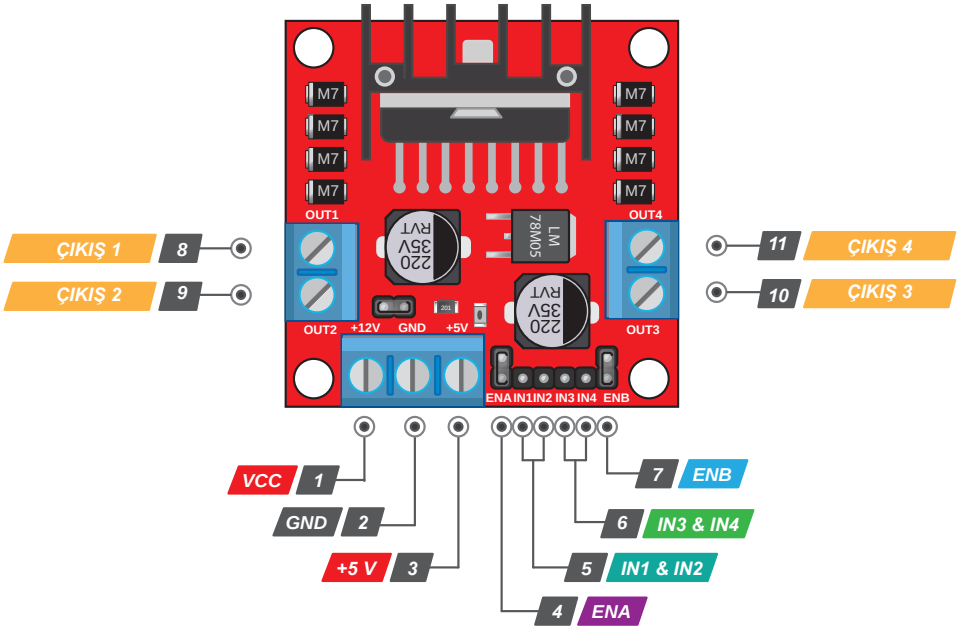


Görsel 3.24: DC motor gerilimini değiştirme (PWM tekniği)

Arduino pininden sağlanan 5 V, 40 mA (0,2 W) değerleri, motorları (mikro DC motorlar hariç) çalıştırmak için yeterli değildir. Bunun için Arduino ile motor arasında motor sürücü adı verilen güç üniteleri kullanılır.

L298N Motor Sürücü

H-Köprüsü oluşturmak için dört ayrı transistör kullanılabildiği gibi entegre devreler de kullanılabilir. Görsel 3.25'te H-Köprüsü içeren L298N motor sürücünün pin yapısı verilmiştir. Benzer özellikleri olan ancak daha küçük boyutlara sahip TB6612FNG motor sürücüsü de uygulamalarda tercih edilebilir.



Görsel 3.25: L298N pin yapısı

VCC: 5 V – 35 V besleme.
GND: Topraklama pini.

5 V: 5 V enable **jumper** yerindeyse bu pin çıkış görevi görür ve Arduino'u beslemek için kullanılır. 5V enable **jumper** çıkmamışsa modül üzerindeki 5 V regülatör pasif bırakılır. L298N içindeki anahtarlama mantığı devresinin gücü kesilir. Bu durumda 5 V pini Arduino üzerindeki 5 V pininden güç alacak şekilde haricî besleme bulunmayan mikro DC motor uygulamasında kullanılabilir.

ENA: A motorunun hız kontrolünü yapar. **Jumper** takılıysa 5 V sağladığından tam güç verilir. **Jumper**ı çıkarılıp bu pin PWM çıkışına bağlandığında A motorunun hızı kontrol edilir.

IN1 ve **IN2:** İki pinin kutuplanmasına göre (lojik 0 veya 1) A motorunun dönüş yönü belirlenir.

IN3 ve **IN4:** İki pinin kutuplanmasına göre (lojik 0 veya 1) B motorunun dönüş yönü belirlenir.

ENB: B motorunun hız kontrolünü yapar. **Jumper** takılıysa 5 V sağladığından tam güç verilir. **Jumper**ı çıkarılıp bu pin PWM çıkışına bağlandığında B motorunun hızı kontrol edilir.

OUT1-OUT2: A motorunun bağlandığı pinler (Tam güçte VCC kadar voltaj çıkışı verir. Çıkış akımı en fazla 2 A olur.).

OUT3-OUT4: B motorunun bağlandığı pinler.

NOT

ENA ve ENB **jumper**ları takılıyken IN1, IN2, IN3 ve IN4 girişleri Arduino PWM çıkışlarına bağlanarak analogWrite() fonksiyonuyla da kullanılabilir. Bu kullanımda kod ve bağlantıdan tasarruf edilir.

Örnek: L298N motor sürücünün IN1, IN2, IN3 ve IN4 girişlerinde analogWrite() kullanımı.

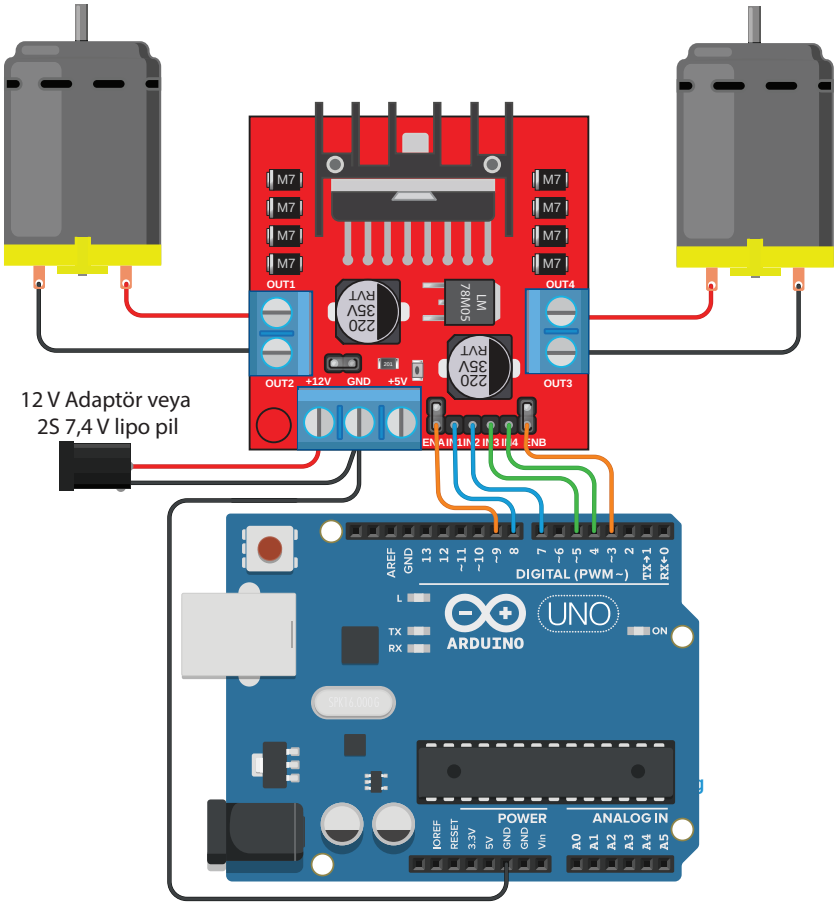
```
const byte IN1 = 5, IN2 = 6, IN3 = 9, IN4 = 10; //Tümü PWM çıkışı.
//enA ve enB jumperları kaldırılmamıştır.
void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}

void loop() {
  analogWrite(IN1, 200);
  analogWrite(IN2, 100);
  analogWrite(IN3, 200);
  analogWrite(IN4, 100);
}
```

Uygulama Adı: DC Motor Uygulaması No.: 8

Amaç: DC motor uygulaması yapmak.

Görsel 3.26'daki devredeki DC motorlar, L298N motor sürücüsüyle sürülmektedir. Uygulamada motorlar birer saniye aralıklarla ileri, geri, sola ve sağa döndürülmektedir. Motorlara uygun tekerlek ve şase takılarak sonuçlar gözlemlenebilir.



Görsel 3.26: L298N'le DC motor sürme

DC motor sürme uygulaması programı aşağıdaki gibidir:

```
const byte IN1 = 8, IN2 = 7, IN3 = 5, IN4 = 4;
const byte enA = 9, enB = 3;
byte hiz = 255; // PWM.
void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}
```

```
void loop() {
  // İleri.
  digitalWrite(IN1, 1);
  digitalWrite(IN2, 0);
  digitalWrite(IN3, 1);
  digitalWrite(IN4, 0);
  analogWrite(enA, hiz); // Tam güç.
  analogWrite(enB, hiz);
  delay(1000);
  // Geri.
  digitalWrite(IN1, 0);
  digitalWrite(IN2, 1);
  digitalWrite(IN3, 0);
  digitalWrite(IN4, 1);
  analogWrite(enA, hiz / 4);
  analogWrite(enB, hiz / 4);
  delay(1000);
  // Sol.
  digitalWrite(IN1, 0);
  digitalWrite(IN2, 1);
  digitalWrite(IN3, 1);
  digitalWrite(IN4, 0);
  analogWrite(enA, hiz / 2); //Yarım güç.
  analogWrite(enB, hiz / 2);
  delay(1000);
  // Sağ.
  digitalWrite(IN1, 1);
  digitalWrite(IN2, 0);
  digitalWrite(IN3, 0);
  digitalWrite(IN4, 1);
  analogWrite(enA, hiz / 2);
  analogWrite(enB, hiz / 2);
  delay(1000);
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.26'daki devreyi kurunuz. Programı yükleyiniz.
4. Motorların dönüş yönünü gözlemleyiniz.
5. Yön sürelerinde ve hızlarında değişiklik yaparak farkı gözlemleyiniz.
6. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Görsel 3.26'daki uygulamada motorları bir saniye durdurmak için komutlar nasıl düzenlenir? Belirtiniz.
2. A0 ve A1 analog girişlerine potansiyometre veya **joystick** bağlayarak dört yön kontrolü yapınız.

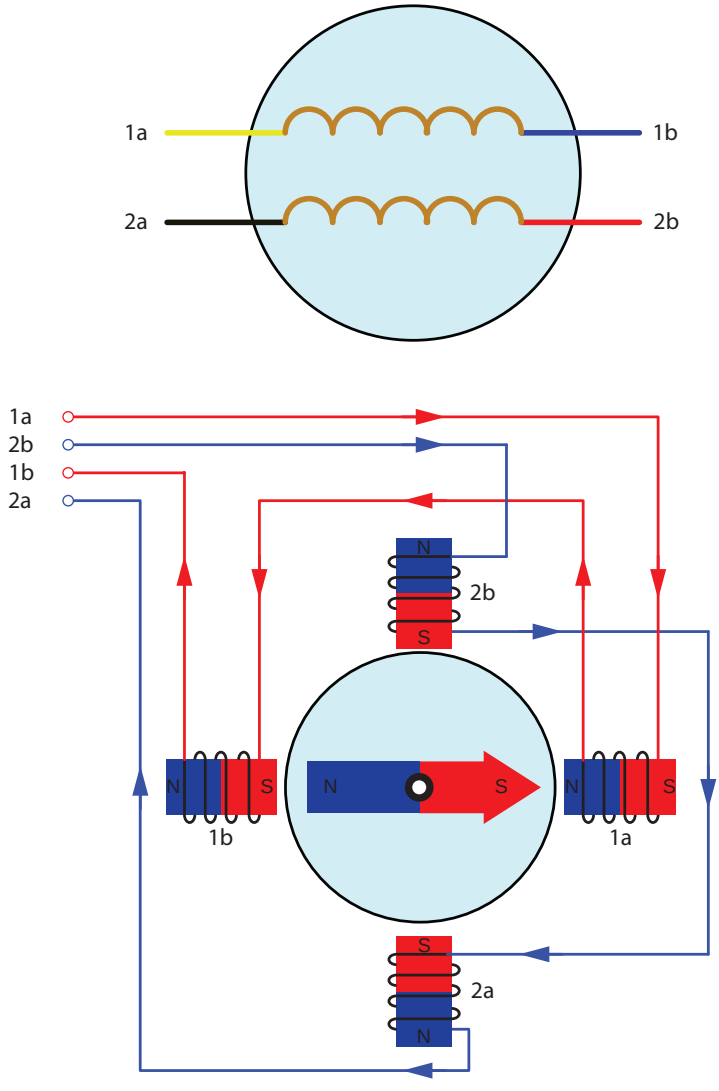
```
const byte IN1 = 5, IN2 = 6, IN3 = 9, IN4 = 10; // Tümü PWM çıkışı.
// enA ve enB jumperları kaldırılmamıştır.
void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}
void loop() {
  byte ileri = map(analogRead(A0), 0, 460, 0, 255); // 460-564 arası dur.
  byte geri = map(analogRead(A0), 564, 1023, 0, 255);
  byte sol = map(analogRead(A1), 0, 460, 0, 255);
  byte sag = map(analogRead(A1), 564, 1023, 0, 255);
  // İleri.
  analogWrite(IN1, ileri);
  analogWrite(IN2, 0);
  analogWrite(IN3, ileri);
  analogWrite(IN4, 0);
  // Geri.
  analogWrite(IN1, 0);
  analogWrite(IN2, geri);
  analogWrite(IN3, 0);
  analogWrite(IN4, geri);
  // Sol.
  analogWrite(IN1, 0);
  analogWrite(IN2, sol);
  analogWrite(IN3, sol);
  analogWrite(IN4, 0);
  // Sağ.
  analogWrite(IN1, sag);
  analogWrite(IN2, 0);
  analogWrite(IN3, 0);
  analogWrite(IN4, sag);
}
```

Sorular

1. Dinamik frenleme (durdurma) nedir? Araştırınız.
2. Dinamik frenleme Görsel 3.26'daki uygulamada nasıl uygulanır? Belirtiniz.
3. enA = 9, enB = 3 pinleri setup fonksiyonunda neden çıkış olarak tanımlanmamıştır? Belirtiniz.

Step Motor Uygulaması

Step motorlar, adımlarla hareket eden DC motorlardır (Görsel 3.27). Faz adı verilen gruplar hâlinde organize edilmiş çoklu bobinleri vardır. Her faza sırayla enerji verildiğinde motor her seferinde bir adım olacak şekilde döner. Çift kutuplu step motorlar, iki bobinden oluşur ve genellikle bobin başına iki olmak üzere dört bağlantıya sahiptir. Tek kutuplu bir step motor elektriksel olarak iki ayrı bobinden oluşur. Her bobinin ortadan bağlantısı olduğu için her bobinde üç bağlantı vardır. Bobindeki orta uçların hepsi birleştiğinden bobin beş bağlantıya sahiptir.

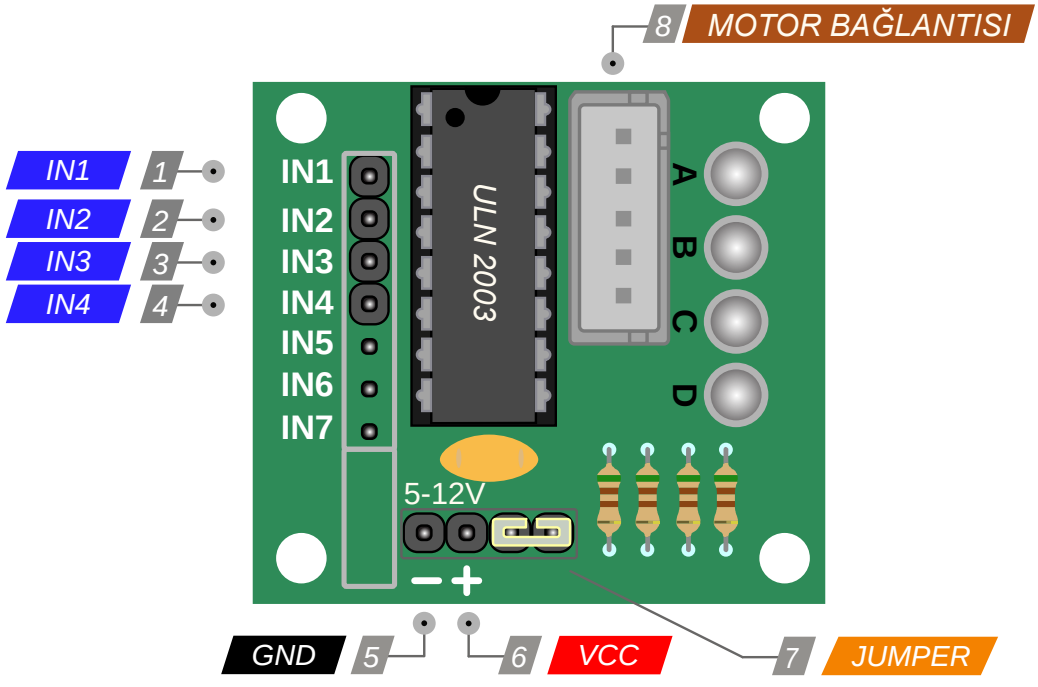


Görsel 3.27: Bipolar step motorun kavramsal modeli

Uygulama 9'da Kullanılan Step Motor ve Sürücü Kartın Özellikleri

28BYJ-48 Step Motor: 5 voltta çalışan 5 telli tek kutuplu bir step motordur. Motorun güç tüketimi 240 mA civarındadır. Motor tam adım modunda çalıştığında her adım 11,25°'lik bir dönüşe karşılık gelir. Bu durum devir başına 32 adım olduğu anlamına gelir (360°/11,25° = 32). Ayrıca motorda 1/64 redüksiyon dişli seti bulunur. (Aslında 1/63,68395) Bunun anlamı, devir başına aslında 32 x 63,68395 =~ 2038 adım demektir.

ULN2003 Sürücü Kartı: ULN2003, yedi darlington transistör çiftinden oluşan motor sürücü kartıdır. Her bir çift 500 mA ve 50 V'a kadar yükleri çalıştırabilir. Kartın, 28BYJ-48 step motor kablolarına uygun konnektörü vardır. Kartta, kontrol giriş hattında adım durumunu gösteren dört adet LED bulunur. Görsel 3.28'de ULN2003 sürücü kartı pin yapısı görülmektedir.



Görsel 3.28: ULN2003 sürücü kartı pin yapısı

IN1-IN4: Motoru sürmek için Arduino'daki dijital çıkış pinlerine bağlanır.

GND: Topraklama pini.

VCC: 5 V-12 V haricî besleme.

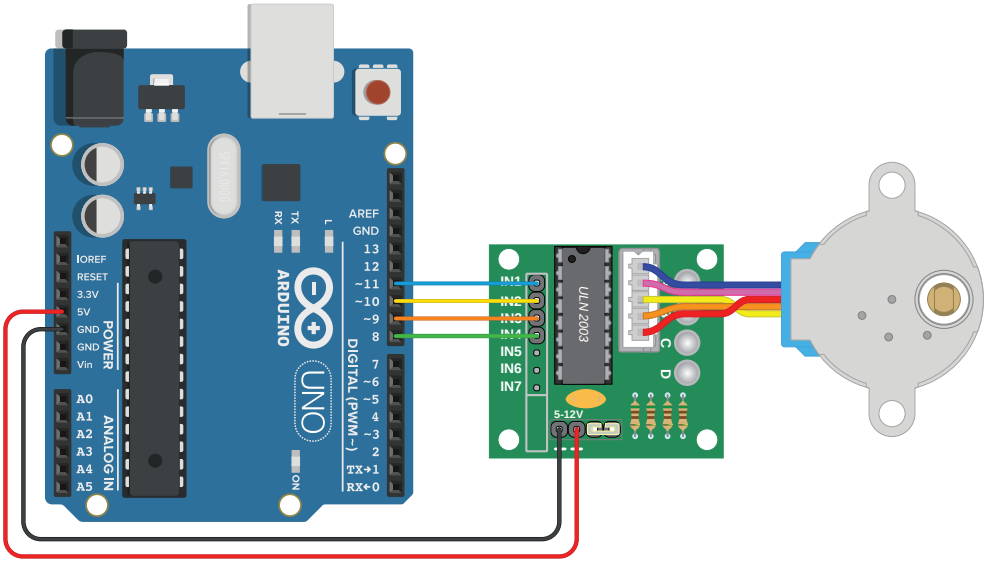
Motor Balants: Step motor konnektörün bağladığı yerdir (Tek yönlü bağlanabilir.).

Jumper: Jumper çıkarıldığında motora enerji gitmez.

Uygulama Adı: Step Motor Uygulaması No.: 9

Amaç: Step motor uygulaması yapmak.

Görsel 3.29'daki uygulamada step motor önce saat yönünde bir tur, sonra saat yönünün tersinde bir tur dönmektedir. Step motor başlığına çubuk benzeri bir aparat takılarak dönüşler daha iyi gözlemlenir.



Görsel 3.29: Step motor uygulaması

Step motor uygulaması programı aşağıdaki gibidir:

```
int x, adimSayisi = 2038, sure = 2;
void setup() {
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
}
void loop() {
  for (x = 0; x < adimSayisi/4; x++) {
  for (byte i = 8; i <= 11; i++) { //Sağa dön.
   digitalWrite(i, 1);
   delay(sure);
   digitalWrite(i, 0);
  }
  }
  for (x = 0; x < adimSayisi/4; x++) {
  for (byte i = 11; i >= 8; i--) { //Sola dön.
 digitalWrite(i, 1);
  delay(sure);
```

```
   digitalWrite(i, 0);
  }
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.29'daki devreyi kurunuz. Programı yükleyiniz.
4. Step motorun çalışmasını gözlemleyiniz.
5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Step motoru iki tur sağa, bir tur sola döndürünüz.
2. Aşağıdaki kod step motoru nasıl çalıştırır?

```
#include <Stepper.h>

const int adimSayisi = 200;  // Bir devirdeki adım sayısı

Stepper stepMotor(adimSayisi, 8, 9, 10, 11);

void setup() {
  stepMotor.setSpeed(60); //Hız 60 rpm ayarlandı.
  Serial.begin(9600);
}

void loop() {
  Serial.println("Sağa dön.");
  stepMotor.step(adimSayisi);
  delay(500);

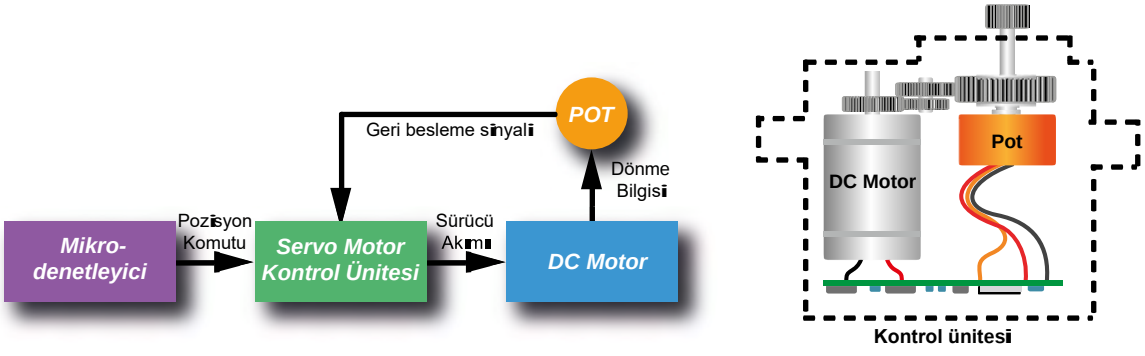
  Serial.println("Sola dön.");
  stepMotor.step(-adimSayisi);
  delay(500);
}
```

Soru

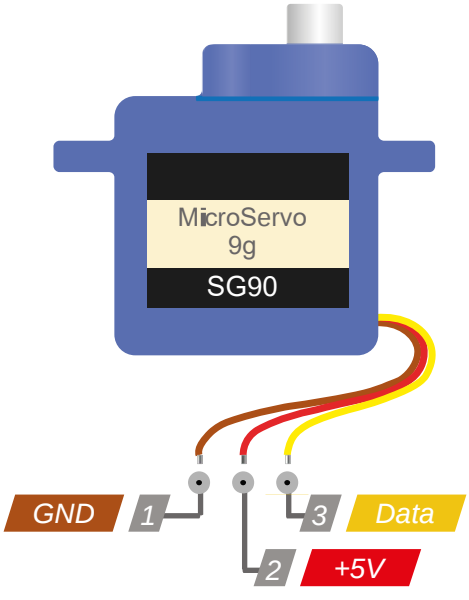
1. 90°lik dönüşte step motor kaç adım atar?

Servo Motor Uygulaması

Servo, kapalı çevrim (geri besleme) kontrol sistemi anlamına gelir. Kapalı çevrim kontrol sistemi, istenen sonucu elde etmek yani motorun hızını ve yönünü ayarlamak için geri besleme sinyalini kullanır. Görsel 3.30'da servo motor geri besleme blok şeması ve kontrol ünitesi görülmektedir. DC motor, dişliler aracılığıyla çıkış miline bağlıdır ki bu mil potansiyometrenin milidir. Potansiyometre, motorun mevcut konumunun hedef konumla karşılaştırıldığı servo kontrol ünitesine konum geri bildirimi sağlar. Hataya göre kontrol ünitesi motorun gerçek konumunu, hedef konumla eşleşecek şekilde düzeltir. Görsel 3.31'de ise mikro servo pin yapısı görülmektedir.



Görsel 3.30: Servo motor geri besleme blok şeması ve kontrol ünitesi



Görsel 3.31: Mikro servo pin yapısı

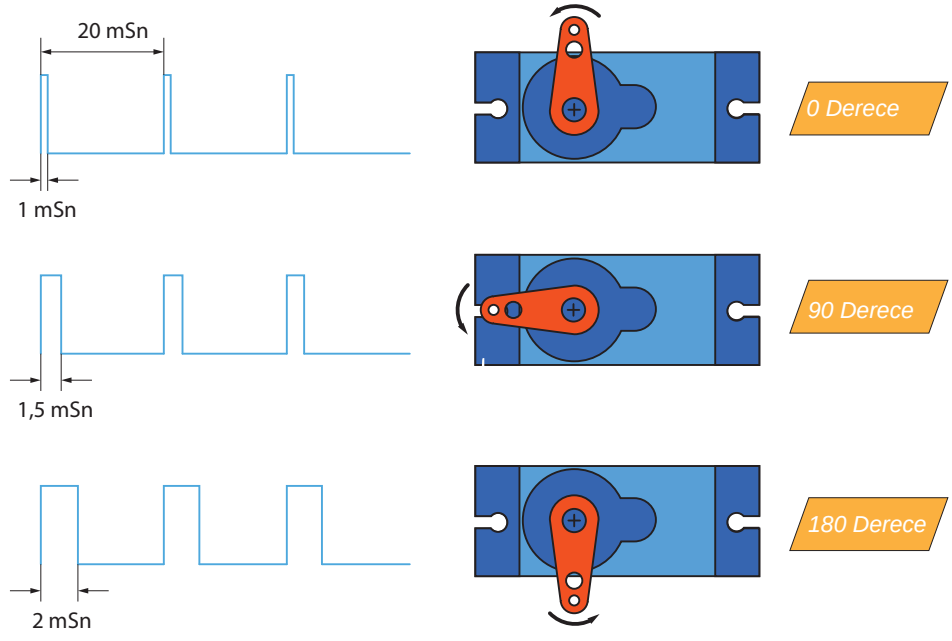
VCC: 5 V besleme.

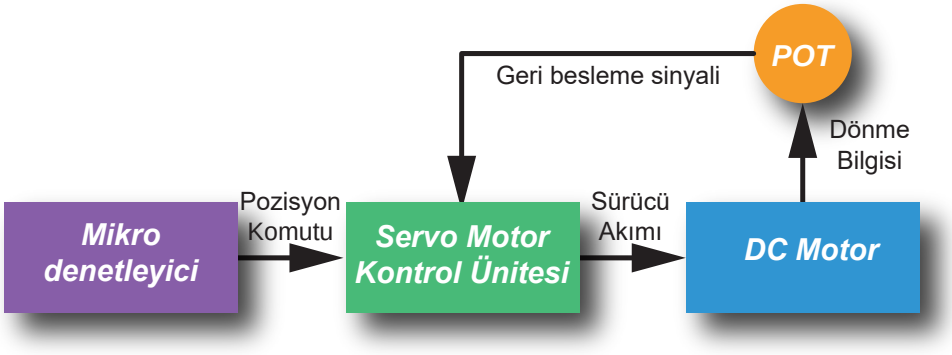
DATA: PWM girişi

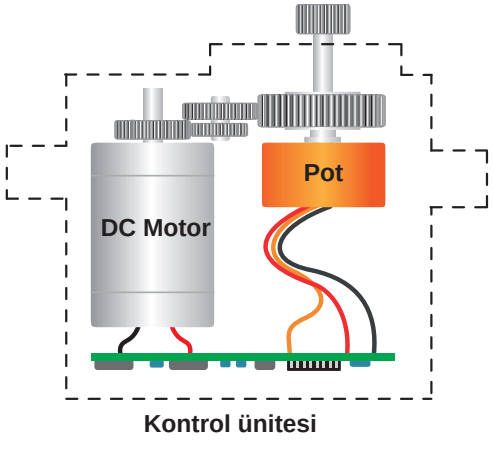
GND: Topraklama pini.

Data hattına PWM sinyali gönderilerek servo motoru kontrol edilir. Görsel 3.32'deki servo motor darbe genişliği ile açı ilişkisinde görüldüğü gibi 20 milisaniyede bir (50 Hz) gönderilen darbenin genişliğine göre açı pozisyonu belirlenir.

Darbe genişliği (lojik 1'de kalma süresi) 1 ms ile 2 ms arasındaki değişim servo motorda 0° ile 180° arasına değişimle eşleşir. Servo motor boşta 10 mA, hareket hâlindeyken 100 mA-250 mA arası akım çeker.







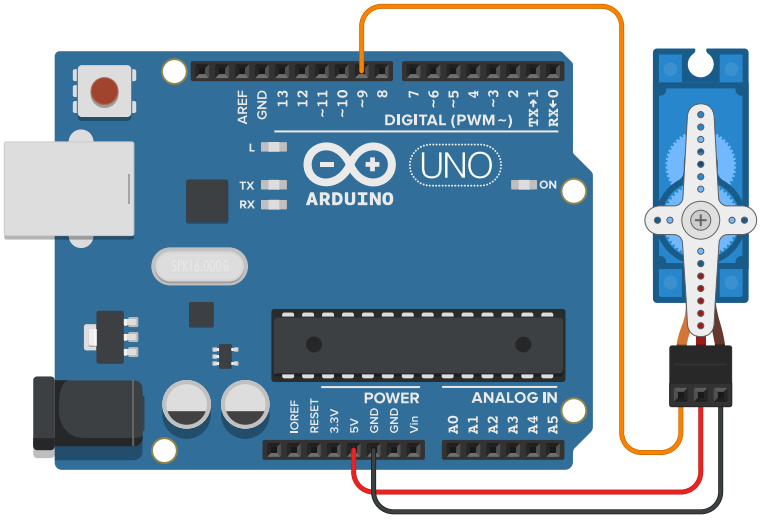
Görsel 3.32: Servo motor darbe genişliği ile açı ilişkisi

Uygulama Adı: Servo Motor Uygulaması — No.: 10

Amaç: Servo motor uygulaması yapmak.

Görsel 3.33'teki uygulamada servo motor 0°den 180°ye birer derece açıyla gidip aynı şekilde geri döner. 0° ile 180° arası mesafe, bu servo motorda 300 ms sürmektedir. 300 ms'den hızlı komutlarda servo motor doğru çalışmaz.

Menülerden "Taslak → Library Ekle → Servo" seçilerek programın başına eklenir. Bu programların örneğine menülerden "Dosya → Örnekler → Servo → Sweep ve Knob" adımları izlenerek de ulaşılabilir.



Görsel 3.33: Mikro servo uygulaması

Mikro servo uygulaması programı aşağıdaki gibidir:

```
#include <Servo.h> // v1.1.7
Servo servo; // Servo isimli nesne oluşturuldu.
byte derece = 0; // Servo derecesini tutan değişken.

void setup() {
  servo.attach(9); // Servo motor 9 numaralı PWM çıkışına bağlı.
}
void loop() {
  for (derece = 0; derece <= 180; derece += 1) { // 0°'den 180°'ye birer derece git.
   servo.write(derece); // Servo motorun gideceği derece.
    delay(15); // 15 ms bekle.
  }
  for (derece = 180; derece >= 0; derece -= 1) { // 180°'den 0°'ye birer derece git
  servo.write(derece);
  delay(15); // 15 x 180 = 2700 ms'de 0°'den 180°'ye tamamlar.
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.33'teki devreyi kurunuz. Programı yükleyiniz.
4. Servo motorun çalışmasını gözlemleyiniz.
5. delay(15); komut satılarını // yorum satırına dönüştürerek programı tekrar yükleyiniz. Programın çalışmasındaki değişikliği gözlemleyiniz.
6. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. A0 girişine potansiyometre veya **joystick** bağlayarak servo motorun pozisyonunu kontrol eden uygulamayı gerçekleştiriniz.

```
#include <Servo.h>

Servo servo; // Servo isimli nesne oluşturuldu.

void setup() {

  servo.attach(9); // Servo motor 9 numaralı PWM çıkışına bağlı.

}

void loop() {

  int pot = analogRead(A0); // Potansiyometrenin konumunu oku. (0 - 1023)

  byte derece = map(pot, 0, 1023, 0, 180);/* Servo motorun kullanacağı 0 - 180 aralığına dönüştür. */

  servo.write(derece); // Servo motorun gideceği derece.

  delay(15);

}
```

Soru

1. Mikro servo motor 1°lik açıyı kaç saniyede tamamlar? Belirtiniz.

3.6. HABERLEŞME UYGULAMALARI

Arduino ve benzeri birçok mikrodenetleyici sıklıkla UART, I2C ve SPI haberleşme standartlarını kullanmaktadır.

Seri İletişim: Bitlerin verinin tek bir iletim yolu üzerinden sırasıyla aktarılmasıdır.

Paralel İletişim: Verinin tüm bitlerinin aynı anda transfer edilmesidir. Paralel veri iletiminde iletilen bilginin her biti için ayrı bir yol (bus) bağlantısı vardır.

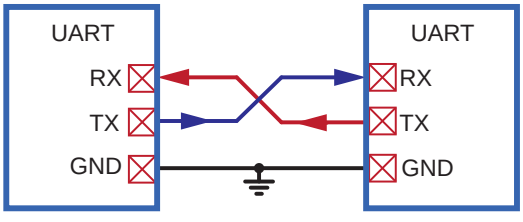
Senkron İletişim: Gönderici clock sinyalinin her yükselen / düşen kenarında verilerle birlikte bir saat sinyali gönderir ve veri değeri alıcı tarafından okunur.

Asenkron İletişim: Saat sinyali yoktur. Alıcı ve gönderici önceden belirlenmiş bir hızda (baud hızı) iletişim kurmaktadır.

Uart/Usart

Arduino ve benzeri birçok mikrodenetleyici üzerinde bulunan seri iletişim birimidir. Bu birimler vasıtasıyla seri iletişim oluşturulur (Görsel 3.34). **UART** [Universal Asynchronous Receiver Transmitter (yunıvırsıl eysingkrınıs risivır transmitır)] Evrensel asenkron alıcı-verici seri iletişim türüdür. **USART** [(Universal Synchronous and Asynchronous Receiver Transmitter (yunıvırsıl singkrınıs end eysingkrınıs risivır transmitır)] ise hem senkron hem de asenkron seri iletişim türüdür.

UART iletişimde veriler **byte** hâlinde iletilir. Her bir **byte** için bir start bir de stop bitleri eklenir. Bir **byte** verinin iletilebilmesi için 10 bit gönderilir. Arduino programlanırken UART kullanılır. Setup() fonksiyonu içerisinde Serial.begin(9600) fonksiyonuyla iletişim hızı (**baudrate**) ayarlanır. Seri haberleşmede iletişim hızı "baud" adı verilen değerle ifade edilir. Bu değer saniyede gönderilen bit sayısını (bps - bits per second) ifade eder. Alıcı ve verici tarafında aynı hız ayarlanmalıdır.



Görsel 3.34: UART haberleşme

I²C Protokolü

I²C, Inter Integrated Circuit (intır intıgreytıd sekıt)] ifadesinin kısaltmasıdır. Aynı zamanda "IIC" olarak da adlandırılır. I²C, düşük hızlı iki hatla kullanılan bir seri protokoldür. Güvenilir iletimin en fazla mesafesi hız arttıkça azalır. En düşük hızda (100 Kbaud veya 100 KHz saat hızı) en fazla mesafe yaklaşık bir metredir.

I²C ve bazı daha yüksek hız modlarına sahiptir. Tüm I²C cihazları şu modları desteklemez:

Hızlı Mod-400 KHz.

Yüksek Hızlı Mod-3,4 MHz.

Ultra Hızlı Mod-5 MHz.

I²C veri yolu, VCC ve GND bağlantısıyla birlikte iki sinyale sahiptir:

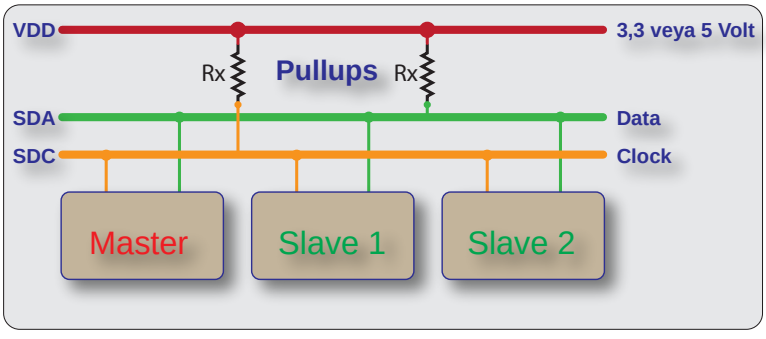
SDA (serial data)-Çift yönlü veri hattıdır.

SCL (serial clock)-Saat sinyalidir.

Her bir sinyal hattına bağlı iki adet pull-up rezistörü vardır (Görsel 3.35). Arduino türlerine göre SDA ve SCL pinleri Tablo 3.2'de gösterilmiştir.

Tablo 3.2: Arduino Türlerine Göre SDA ve SCL Pinleri

Arduino türü	SDA pini	SCL pini
Arduino Uno	A4	A5
Arduino Mega	20	21
Arduino Leonardo	2	3
Arduino Due	20	21
Arduino Nano	A4	A5



Görsel 3.35: I²C haberleşme

I²C veri yoluna ara birim oluşturan **master** ve **slave** isimli iki tür cihaz vardır. **Master** cihaz veri yolunu kontrol eder ve saat sinyalini sağlar. **Slave** cihazlardan tek tek veri ister. Veri yolu üzerinde birden fazla **master** cihaz olabilir ancak herhangi bir anda sadece bir tanesi aktif **master** olabilir. **Master** cihazların kendilerine atanmış bir adresi yoktur. Slave cihazların bir adresi vardır ve bu adresin veri yolunda benzersiz olması gerekir. 7 bitlik adresleme kullanıldığından I²C veri yolunda 128'e kadar slave cihaz olabilir. Yeni nesilde 10 bitlik adresleme de kullanılmaktadır.

Arduino'da I2C protokolüyle haberleşmek için yerleşik Wire.h kütüphanesi kullanılır.

start() - Kütüphaneyi başlatır ve Arduino'yu **master** veya **slave** olacak şekilde ayarlar.

requestFrom() - **Master** tarafından bir **slave**dan veri istemek için kullanılır.

startTransmission() - **Master** tarafından belirtilen bir **slave**a veri göndermek için kullanılır.

endTransmission() - **Master** tarafından startTransmission fonksiyonuyla başlatılan bir iletimi sonlandırmak için kullanılır.

write() - I2C veri yoluna veri göndermek için hem **master** hem de **slave** tarafından kullanılır.

available() - Alınan verinin **bayt** sayısını belirlemek için hem **master** hem de **slave** tarafından kullanılır. read() - I2C veri yolundan bir **bayt** veri okur.

SetClock() - **Master** tarafından belirli bir saat frekansını ayarlamak için kullanılır.

onReceive() - **Master** cihazdan veri geldiğinde **slave** cihaz tarafından bir fonksiyon çalıştırılır.

onRequest() - **Master** cihaz veri istediğinde **slave** cihaz tarafından bir fonksiyon çalıştırılır.

SPI Protokolü [Serial Peripheral Interface (siriıl pırifrıl intırfeys)]

Senkron haberleşme protokollerinden bir tanesidir. **Full duplex** (eş zamanlı çift yönlü

çalışabilen) olarak çalışabilir. Haberleşme gerçekleştirilecek cihazlar arasında **master - slave** ilişkisi vardır. Birden fazla **slave** cihazla haberleşme sağlanabilir. Senkron olarak çalıştığı için clock sinyali kullanılır. Kısa mesafeli iletimde tercih edilir (Görsel 3.36).

Master ve çevresel cihazlara bağlanan **MISO** (Master In Slave Out), **MOSI** (Master Out Slave In) ve **SCK** (Serial Clock) olmak üzere üç adet SPI hattı bulunur.

MISO: Çevresel cihazlardan (**slave**) yollanan verilerin **master** cihaza aktarıldığı hattır.

MOSI: Master cihazdan yollanan verilerin çevresel cihazlara aktarıldığı hattır.

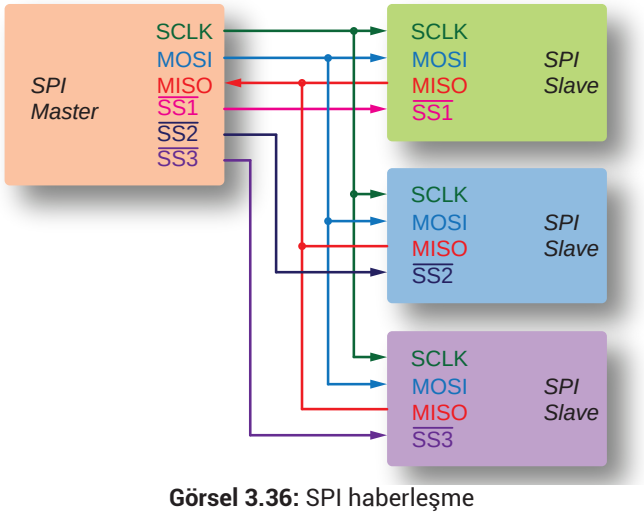
SCK: SPI haberleşmesinde senkronu sağlayan saat sinyalinin bulunduğu hattır. Saat sinyali **master** cihaz tarafından üretilir.

SPI protokolünde I²C'den farklı olarak veri hatları tek yönlüdür. **Slave** cihazların adreslerinin olmasına gerek yoktur. Her **slave** cihazın seçim ayağı bulunur. Bu ayağa **SS** (Slave Select) denir. Bu hattın sayısı kullanılan slave cihazların sayısı kadardır. Her cihaz için **master** cihazından ayrı SS hattı çıkar. SS hattı LOW (0 volt) düzeyinde olan **slave** cihaz, **master** cihazla iletişime başlar.

Arduino türlerine göre MOSI, MISO ve SCK pinleri Tablo 3.3'te gösterilmiştir. SS için herhangi bir dijital çıkış kullanılabilir.

Tablo 3.3: Arduino Türlerine Göre MOSI, MISO ve SCK Pinleri

Arduino türü	MOSI	MISO	SCK
Arduino Uno	11	12	13
Arduino Nano	11	12	13
Arduino Mega	51	50	52



Görsel 3.36: SPI haberleşme

Arduino'da SPI protokolüyle haberleşmek için yerleşik SPI.h kütüphanesi kullanılır.

SPI.begin(): SPI haberleşmesini başlatır ve SPI pinlerini başlangıç konumlarına alır.

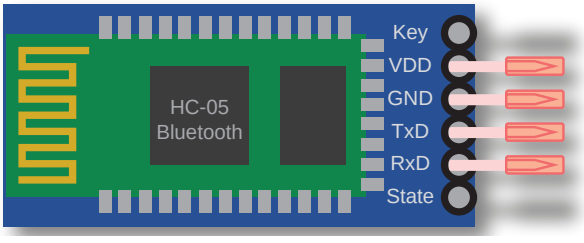
SPI.setClockDivider(): SPI haberleşmesinin saati ayarlanır. Fonksiyon, değer olarak saat değişkenlerini almaktadır. Eğer hiçbir değişiklik yapılmazsa SPI saati "SPI_CLOCK_DIV4" olarak çalışır. Fonksiyonun alabileceği değişkenler; SPI_CLOCK_DIV4, SPI_CLOCK_DIV8, SPI_CLOCK_DIV16,

SPI_CLOCK_DIV32, SPI_CLOCK_DIV64, SPI_CLOCK_DIV128'dir.

SPI.transfer(): SPI hattına veri yollamak veya veri almak için bu fonksiyon kullanılır.

Bluetooth Haberleşme

Bluetooth, veri iletmek ve almak için 10 metre (Bulunduğu ortamdan etkilenir.) menzilli bir kablosuz teknolojidir. 2,400 ila 2,485 GHz arasında ISM (Endüstri, Bilim ve Medikal) bandında çalışır. Bluetooth haberleşme için **UART** protokolü kullanılır. Varsayılan haberleşme hızı 9600 bps'dir. 115,200 **bauda** kadar hızlarda çalışabilir. Modüller ayrıca "AT" metin dizisiyle başlayan standart bir komutlar seti olan AT komut setini de destekler. Görsel 3.37'de HC-05 bluetooth modülünün pin yapısı görülmektedir.



Görsel 3.37: HC-05 pin yapısı

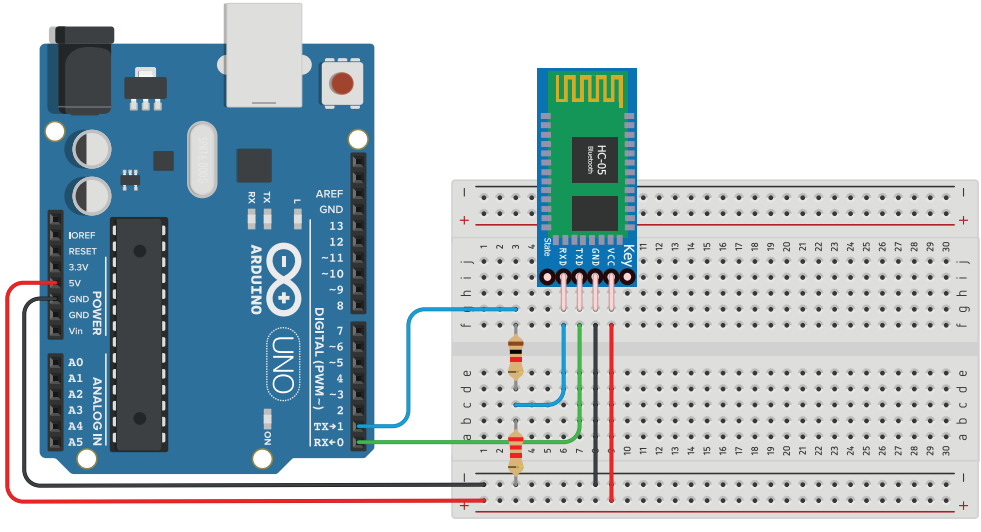
VCC: 3,3 V-5 V besleme.

GND: Topraklama pini.

TXD: Veri çıkışı (3,3 V).

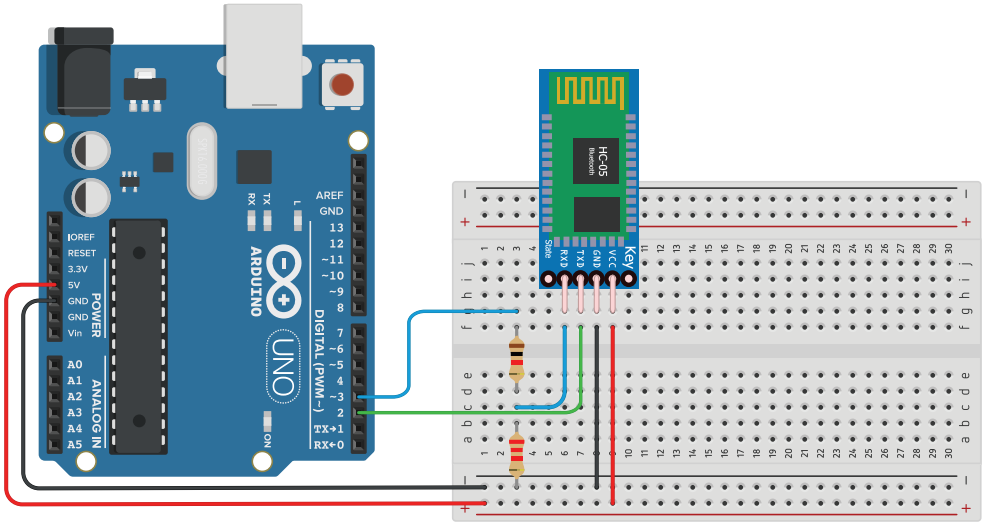
RXD: Veri girişi (3,3 V).

DİKKAT Modülün TX ucu Arduino'nun RX (0. pin) ucuna, modülün RX ucu Arduino'nun TX (1. pin) ucuna bağlanmalıdır. Ancak modülün TX ucundan çıkan 3,3 V Arduino RX ucunda lojik 1 olarak kabul edilirken Arduino'nun TX ucundan çıkan 5 V, modülün RX ucuna doğrudan uygulandığında bozulmasına neden olacaktır. Bu nedenle Arduino'nun TX ucundan çıkan 5 V'u 3,3 V'a düşürmek için Görsel 3.38'deki gibi gerilim bölücü dirençler kullanılmalıdır.



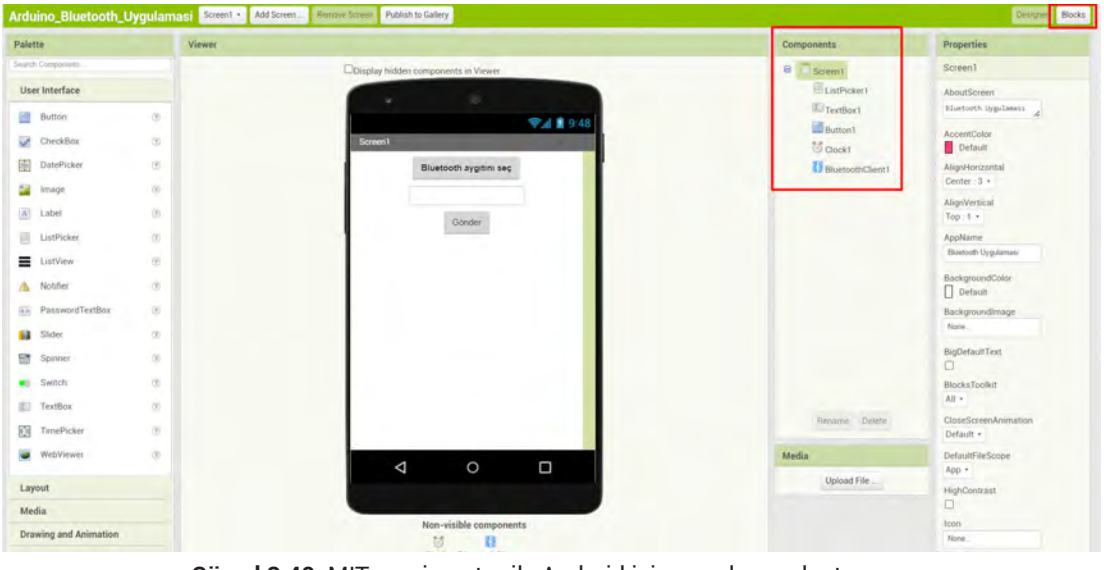
Görsel 3.38: Gerilim bölücü dirençlerle bluetooth modülün Rx ucuna 3,3 V verme

Bluetooth modülü doğrudan Arduino'nun 0. ve 1. pinlerine bağlandığında program yüklenirken (program yükleme bu pinlerden yapıldığı için) hata verir. Modül sökülerek program yüklenir ve modül tekrar takılır. Sök tak işlemini yapmamak için SoftwareSerial.h kütüphanesi kullanılarak istenen dijital pinler (Görsel 3.39'da 2 ve 3 numaralı pinler seçilmiştir.) TX ve RX olarak ayarlanır. Menülerden "Taslak → Library Ekle → SoftwareSerial" seçilerek programın başına eklenir.



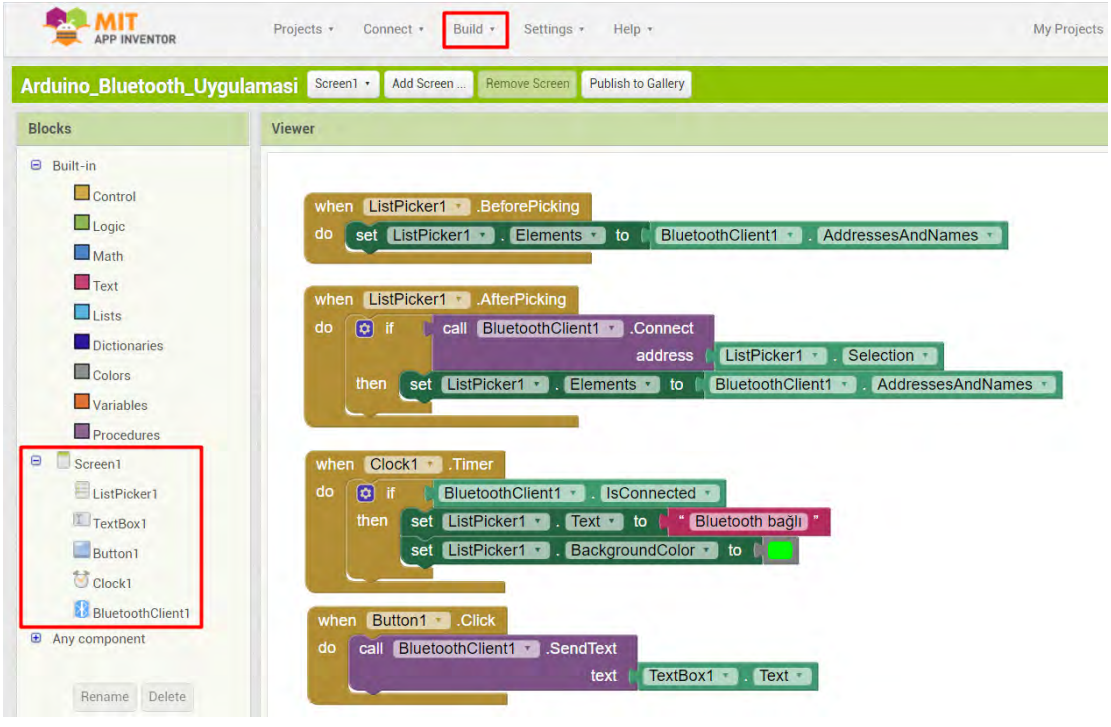
Görsel 3.39: SoftwareSerial.h kütüphanesi ile TX ve RX'i farklı pinlerde kullanma

Devreye enerji verildiğinde bluetooth modülle haberleşmek için Android telefondan bluetooth bağlantısı açılarak hc-05 cihazı bulunur. Varsayılan şifre "1234" veya "0000"dır. Android telefondan veri gönderebilmek için **play store**dan "Arduino bluetooth controller" benzeri uygulamalar kullanılır veya http://ai2.appinventor.mit.edu adresinden basit bir Android uygulaması yapılarak karakterler bluetooth üzerinden gönderilir. Uygulamayı yapmak için Görsel 3.40'taki "Components" bölümünde verilen elemanlar soldaki listeden seçilir. "Bloks" düğmesine basılarak uygulamanın kodları yazılır.



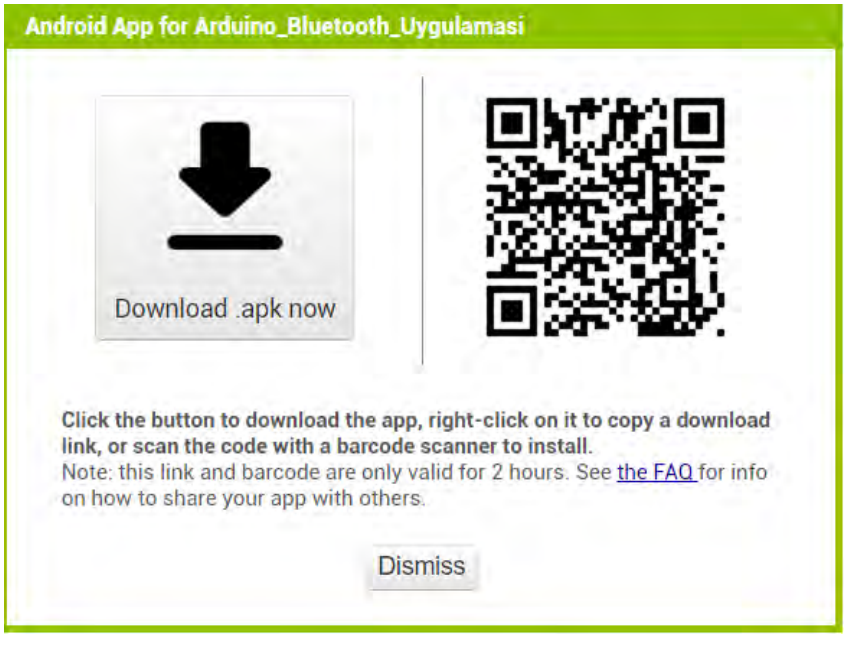
Görsel 3.40: MIT app invertor ile Android için uygulama oluşturma

Görsel 3.41'de görüldüğü gibi soldaki menüden elemanların üzerine tıklanarak ilgili komutlar seçilerek kodlar yazılır. "Build" düğmesine tıklanarak Android App (.apk) oluşturulur.



Görsel 3.41: Uygulama kodları

Görsel 3.42'deki gibi oluşturulan karekod (Kısa bir süre geçerlidir.) okutularak uygulama telefona indirilir. Uygulama yüklenirken "Bilinmeyen kaynaklardan yükle." seçeneği açılmalıdır.



Görsel 3.42: Karekodla akıllı telefona uygulama indirme

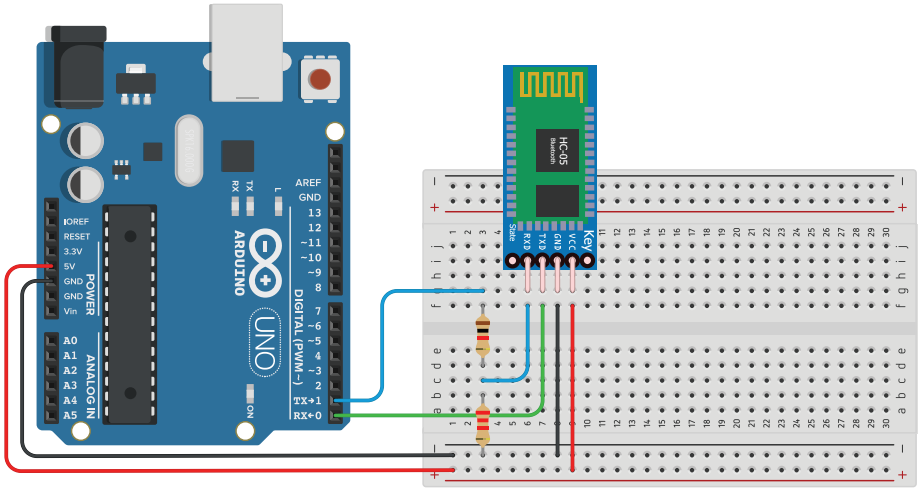
Uygulama Adı: Bluetooth Uygulaması — No.: 11

Amaç: Bluetooth uygulaması yapmak.

Bluetooth modülle haberleşmek için telefondan bluetooth bağlantısı açılarak hc-05 cihazı bulunur ve "Bluetooth aygıtını seç" düğmesiyle hc-05 seçilir. Varsayılan şifre "1234" veya "0000" dır. Bluetooth ile "A" harfi gönderilerek LED yakılır, "B" harfi gönderilerek LED söndürülür (Görsel 3.43). Bluetooth uygulama devresi Görsel 3.44'te görülmektedir.



Görsel 3.43: Uygulamayla bluetooth üzerinden karakter gönderme



Görsel 3.44: Bluetooth uygulaması

Bluetooth uygulaması programı aşağıdaki gibidir:

```
void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600); // Bluetooth'la seri iletişimi başlat.
}
void loop() {
  if (Serial.available()) { // Veri geliyorsa...
  char karakter = Serial.read(); // Gelen veriyi karakter değişkenine ata.
  if (karakter == 'A') // Karakter A ise...  (a farklı bir karakterdir.)
   digitalWrite(13, 1); // LED'i yak.
  else if (karakter == 'B') // Karakter B ise...
   digitalWrite(13, 0); // LED'i söndür.
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.44'teki devreyi kurunuz. Programı yükleyiniz.
4. Telefondan buluetooh açarak modülle eşleştiriniz.
5. Telefondan uygulamayla A karakteri gönderek LED'in yandığını gözlemleyiniz.
6. Telefondan uygulamayla B karakteri gönderek LED'in söndüğünü gözlemleyiniz.
7. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. SoftwareSerial.h kütüphanesini kullanarak PWM çıkışına bağlı LED'in parlaklığını değiştiriniz. Telefondan 0-255 arası sayı gönderiniz.

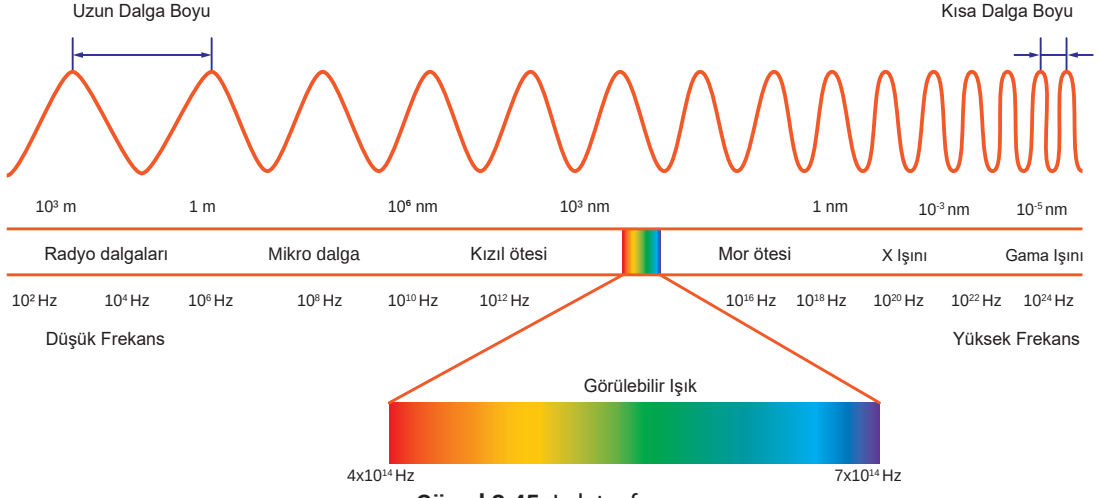
```
#include <SoftwareSerial.h> // Dahili olarak Arduino IDE'de bulunur.
SoftwareSerial bluetooth(10, 11); // Arduino 10. pin RX, 11. pin TX olarak ayarladı.
byte LED = 3; // LED 3 numaralı PWM özellikli pine bağlı.
void setup() {
  bluetooth.begin(9600); // Bluetooth ismiyle oluşturulan nesne başlatıldı.
}
void loop() {
  if (bluetooth.available()) { // 10. pinden veri geliyorsa...
  String veri = bluetooth.readString(); // 0 - 255 arası gelen karakterleri oku.
  int PWM = veri.toInt(); // Gelen metni integer veri tipine dönüştür.
  analogWrite(LED, PWM);
  }
}
```

Soru

1. Gelen veriden hem açma kapama bilgisi hem de PWM bilgisi nasıl ayırt edilerek okunur? Açıklayınız.

Kızılötesi (İnfrared) Işınım ile Haberleşme

Kızılötesi (infrared) ışınım, insan gözünün göremediği dalga boyundadır. Kızılötesi ışınımın dalga boyu 750 nanometre ile 1 mikrometre arasındadır. Görsel 3.45'te ışık tayfı görülmektedir.



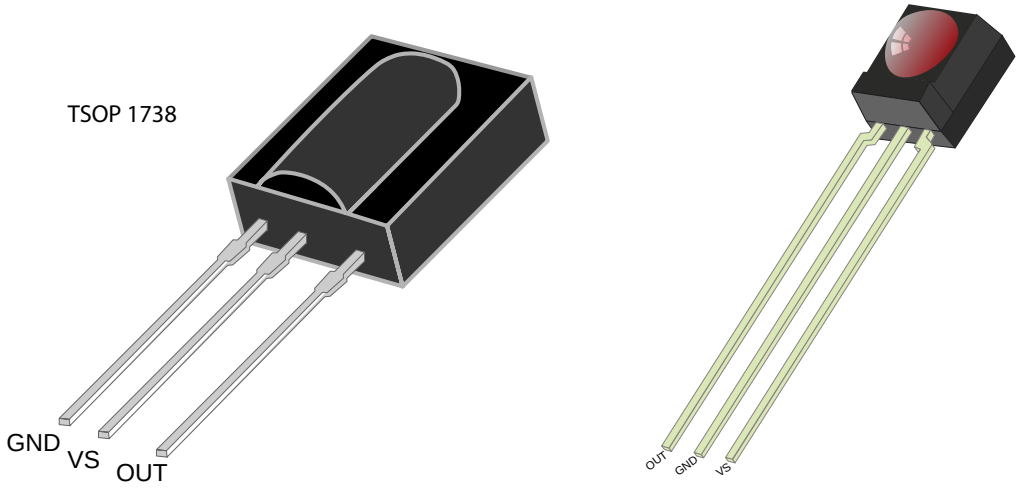
Görsel 3.45: Işık tayfı

Görsel 3.46'da görülen **TSOP1738** 38 KHz IR alıcı modülü üç adet pine sahiptir.

VS (VCC): Besleme gerilimi.

GND: Topraklama pini.

OUT: Çıkış.



Görsel 3.46: IR alıcı pin yapısı

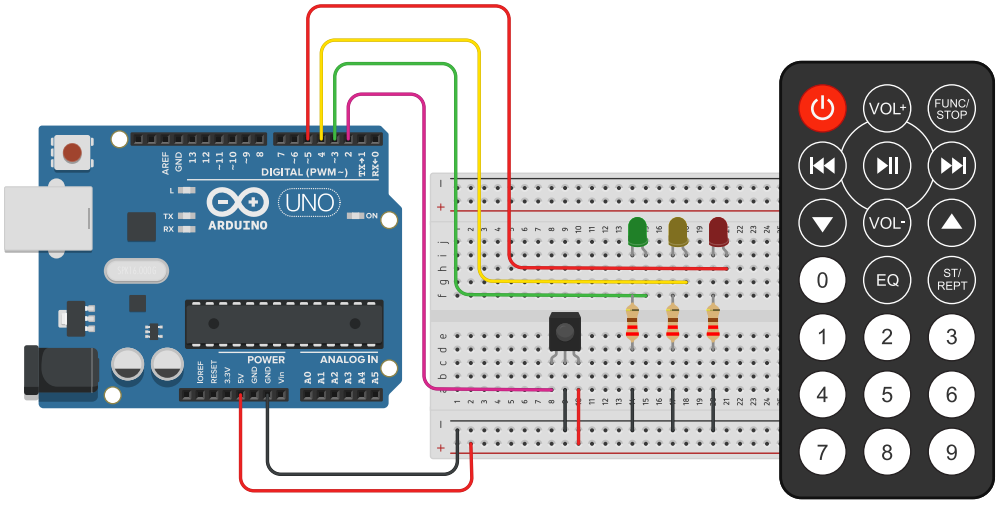
Uygulama Adı: IR ile Uzaktan Kumanda Uygulaması No.: 12

Amaç: IR ile uzaktan kumanda uygulaması yapmak.

Görsel 3.47'de IR ile uzaktan kumanda uygulaması görülmektedir. Burada kullanılan kumandalarda her tuşun farklı bir kodu vardır. Kullanılmak istenen tuşun kodunu öğrenebilmek için menülerden "Taslak → Library Ekle → Kütüphaneleri Yönet" seçeneğinden arama kutusuna "IRremote" yazılarak "IRremote" kütüphanesi bulunur ve kurulur. Menülerden "Taslak → Library Ekle → IRremote" seçilerek programın başına eklenir.

Aşağıdaki program herhangi bir kumandanın basılan tuşlarının **hex**. kodlarını seri ekranda gösterir. Tuş kodları uygulamada kontrol için kullanılmaktadır.

IR alıcının data çıkışı Arduino'nun 2 numaralı pinine bağlıdır. Çıkış olarak ayarlanan pinlere üç ayrı LED veya RGB LED bağlanabilir. Kumandanın 1, 2, 3 numaralı tuşları LED'leri yakıp söndürmektedir. Çıkışta MOSFET, röle vb. güç üniteleri kullanılarak şerit LED, lamba, fan vb. yükler kontrol edilebilir.



Görsel 3.47: IR ile uzaktan kumanda uygulaması

IR ile uzaktan kumanda uygulaması programı aşağıdaki gibidir:

```
#include <IRremote.h>
IRrecv irAlici(2); // irAlici isimli nesne 2. pini giriş olarak kullanır.
decode_results tus; // Kodu çözülmüş sonuçları tus değişkenininde sakla.
#define tus1 0xFF30CF // Burada kullanılacak Hex kodu seri ekranda görünür.
#define tus2 0xFF18E7 #define tus3 0xFF7A85
const byte LED[] = {3, 4, 5};

void setup()
{
  Serial.begin(9600);
  irAlici.enableIRIn(); // Alıcıyı başlat.
  pinMode(LED[0], OUTPUT);
  pinMode(LED[1], OUTPUT);
```

```
  pinMode(LED[2], OUTPUT);
}
void loop() {
  if (irAlici.decode(&tus)) // Tuşa basılmışsa...
  {
  Serial.println(tus.value, HEX); // Tuşun Hex kodunu yazdır.
  if (tus.value == tus1)
   digitalWrite( LED[0], !digitalRead(LED[0])); // Çıkışı tersle.
  if (tus.value == tus2)
   digitalWrite( LED[1], !digitalRead(LED[1])); // Çıkışı tersle.
  if (tus.value == tus3)
   digitalWrite( LED[2], !digitalRead(LED[2])); // Çıkışı tersle.
  irAlici.resume();
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.47'deki devreyi kurunuz. Programı yükleyiniz.
4. Kumandadan 1, 2, 3 numaralı tuşlara basarak LED'leri yakıp söndürünüz.
5. Kumanda tuşlarının kodlarını seri ekrandan görebilirsiniz. Her bir tuş için #define tus1 0xFF 30CF şeklinde satırlar düzenleyerek diğer tuşları da aktif hâle getirebilirsiniz.
6. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. 0. tuşa basıldığında LED'i yakıp söndüren düzenlemeyi gerçekleştiriniz.

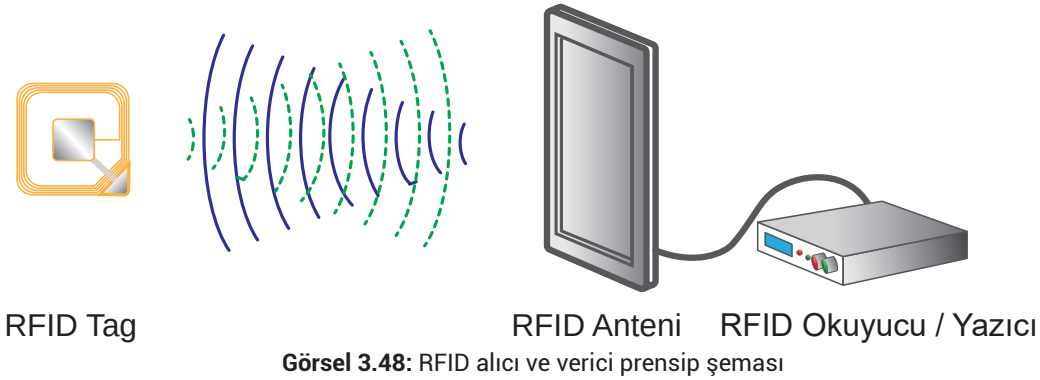
Soru

1. TV kumandaları bu uygulamada kullanılabilir mi? Açıklayınız.

RFID Uygulamaları

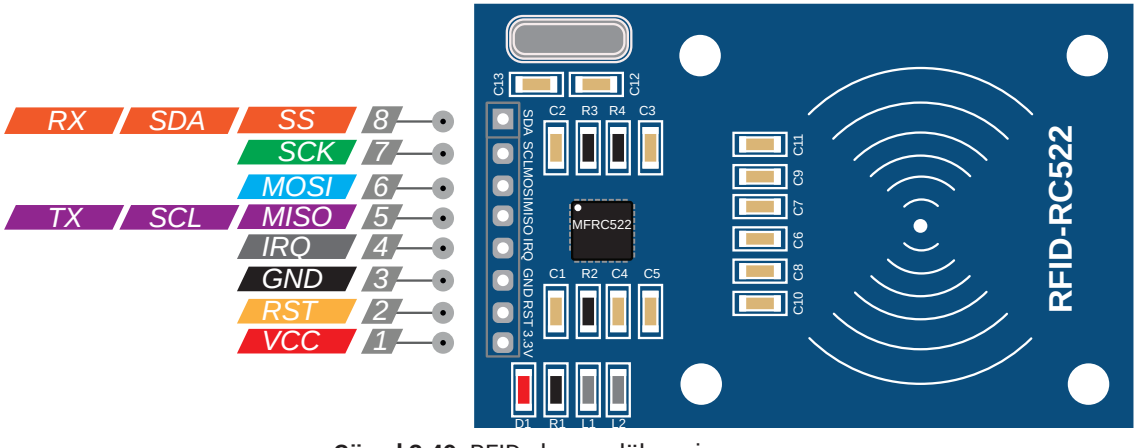
RFID [Radio Frequency IDentification (redyio frikwınsi aydentıfıkeyşın) Radyo Frekans Tanımlama] sistemi, tanımlanacak bir nesneye eklenen bir verici (aktarıcı/etiket) ve alıcı (sorgulayıcı/okuyucu) olmak üzere iki ana bileşenden oluşur.

Alıcı, bir radyo frekansı modülünden ve yüksek frekanslı elektromanyetik alan oluşturan bir antenden meydana gelir. Verici ise genellikle pasif (Pil içermez.) bir cihazdır. Vericinin içinde bilgiyi depolayan ve işleyen bir mikroçip ile sinyali alıp iletmek için bir anten bulunur. Karttaki (verici) kodlanmış bilgilerin okunabilmesi için okuyucunun (alıcı) yakınına yerleştirilir (Okuyucunun doğrudan görüş alanı içinde olması gerekmez.). Okuyucu, elektronların kartın anteninden geçmesini ve ardından çipe güç vermesini sağlayan bir elektromanyetik alan oluşturur. Kartın içindeki güç verilen çip, depolanan bilgiyi başka bir radyo sinyali biçiminde okuyucuya geri göndererek yanıt verir. Görsel 3.48'de RFID alıcı ve verici prensip şeması görülmektedir.



Görsel 3.48: RFID alıcı ve verici prensip şeması

Görsel 3.49'da RC522 RFID alıcı modül ve pin yapısı görülmektedir. RC522 RFID alıcı modülü, RFID etiketleriyle (ISO 14443A standart etiketleri) iletişim kurmak için kullandığı 13,56 MHz'lik bir elektromanyetik alan oluşturmak üzere tasarlanmıştır. Okuyucu, maksimum 10 Mbps veri hızında dört pinli Seri Çevresel Arabirim (SPI) üzerinden bir mikrodenetleyiciyle iletişim kurabilir. Ayrıca I²C ve **UART** protokolleri üzerinden iletişimi destekler.



Görsel 3.49: RFID alıcı modül ve pin yapısı

VCC: 3,3 V besleme. Arduino'nun 3 V çıkışına bağlanır. 5 V pinine bağlamak modüle zarar verir.

RST: Yükselen kenarda modülü sıfırlar.

GND: Topraklama pini.

VCC: 3,3 V besleme. Arduino'nun 3 V çıkışına bağlanır. 5 V pinine bağlamak modüle zarar verir.

RST: Yükselen kenarda modülü sıfırlar.

GND: Topraklama pini.

IRQ: Verici kart yaklaştırıldığında mikrodenetleyiciye uyaran kesme pinidir.

MOSI (Master Out Slave In): RC522 modülüne SPI girişidir.

SCK: SPI için Arduino tarafından sağlanan saat darbelerini kabul eder.

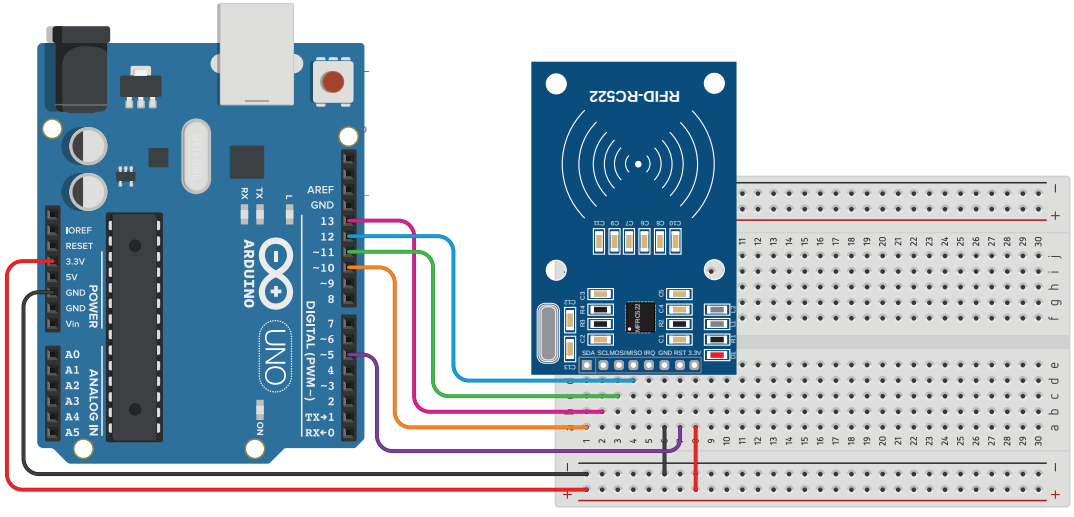
SS / SDA / RX: SPI arayüzü etkinleştirildiğinde sinyal girişi görevi, I^2C arayüzü etkinleştirildiğinde seri veri görevi ve **UART** arayüzü etkinleştirildiğinde seri veri girişi görevi görür.

MİSO / SCL/ TX: SPI arayüzü etkinleştirildiğinde MISO (master in slave out) görevi, I^2C arayüzü etkinleştirildiğinde SCL (serial clock) görevi ve **UART** arayüzü etkinleştirildiğinde seri veri çıkışı görevi görür.

Uygulama Adı: RFID Uygulaması — No.: 13

Amaç: RFID uygulaması yapmak.

Görsel 3.50'deki uygulamada RFID okuyucuya geçerli bir kart okutulduğunda seri ekranda "**geçerli kart**" yazar. Geçersiz kart okutulduğunda kartın ID'sini verir. ID'si programa dâhil edilen kartlar, geçerli kart olur. Kart geçerliyken çalışan kod blokuna istenen kodlar yazılarak istenen kontroller sağlanır.



Görsel 3.50: RFID uygulaması

RFID uygulaması programı aşağıdaki gibidir:

```
#include <SPI.h> // Dahili SPI kütüphanesi
#include <MFRC522.h>  // v1.4.9
#define reset 9 // Reset pini
#define ss 10  // SS pin
MFRC522 kartOkuyucu(ss, reset);  // Kart okuyucu isimli nesne oluşturuldu.
byte i, j, k;
byte kartSayisi = 2; //Kart sayısı doğru girilmelidir.
byte kartlar[][4] = {
  {0xA6, 0x06, 0x1E, 0xF1}, // ilk kartın dört byte'lık ID'si HEX olarak tanımlandı.
  {0xA6, 0x06, 0x1E, 0xF2, // ikinci kart...
};

void setup() {
  Serial.begin(9600);                    //Bilgisayarın seri ekranıyla UART iletişimi.
  SPI.begin();                           // MFRC522 kart okuyucusuyla SPI iletişimi.
  kartOkuyucu.PCD_Init();                // MFRC522'yi başlat
 Serial.println("Lütfen kartı okutunuz.");
}
void loop() {
```

```
  if ( ! kartOkuyucu.PICC_IsNewCardPresent())  //Bir kart okunana kadar bekle.
  return; //Kart yoksa başa dön.
  if ( ! kartOkuyucu.PICC_ReadCardSerial())  // Kartı oku.
  return;
  if (kartKontrol()) { // Kart geçerliyse yapılacaklar...
  Serial.println("Kart geçerli.");
  }
  else { // Kart geçerli değilse yapılacaklar...
  for (i = 0; i < 4; i++) {
   Serial.print(kartOkuyucu.uid.uidByte[i], HEX); //Geçersiz kartın ID'sini ekrana yazdır.
   Serial.print(" ");
  }
  Serial.println(" numaralı kartınız sisteme tanımlı değil!");
  }
  delay(500);
  Serial.println("\nLütfen kartı okutunuz.");
}

boolean kartKontrol() { // Kart geçerliye 1, değilse 0 döndürür.
  k = 0;
  for (j = 0; j < kartSayisi; j++) { // Kayıtlı tüm kartlara bak.
    for (i = 0; i < 4; i++) { // Her kartın ID'si dört byte.
  if (kartlar[j][i] == kartOkuyucu.uid.uidByte[i]) {//Okunan kartı kayıtlı olanlarla karşılaştır.
        k++;
        if (k == 4) { // Tüm byte'lar doğrulandıysa...
          return 1; // 1 döndür.
        }
      }
    }
  }
  return 0; // Okunan kart kayıtlı kartlarla eşleşmediyse 0 döndür.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Görsel 3.50'deki devreyi kurunuz. Programı yükleyiniz.
3. Seri port ekranını açınız.
4. Kartı okutunuz. Kartın ID'si **hex.** formatında ekranda yazacaktır.
5. Kartın **hex.** kodlarını programdakiyle değiştiriniz.
6. Geçerli kartı okutarak seri ekranı gözlemleyiniz.

DİKKAT D'deki **byte**ları yazdırırken 16'dan küçük sayıları ekrana tek haneli yazar. Örneğin **byte**ın değeri 06 ise 6 şeklinde yazar. ID her **byte** arasına boşluk bırakılmadan yazdırılırsa ID yanlış tanımlanabilir.

SIRA SİZDE

1. Geçerli kart okutulunca servo motorla kapı açan uygulamayı gerçekleştiriniz.

```
#include <SPI.h> // Dahili SPI kütüphanesi
#include <MFRC522.h>  // v1.4.9
#include <Servo.h> // Dahili Servo kütüphanesi v1.1.8

Servo servo; // Servo isimli nesne oluşturuldu.

#define reset 9 // Reset pini
#define ss 10  // SS pin

MFRC522 kartOkuyucu(ss, reset);  // Kart okuyucu isimli nesne oluşturuldu.

byte i, j, k;
byte kartSayisi = 2; //Kart sayısı doğru girilmelidir.
byte kartlar[][4] = {
  {0xA6, 0x06, 0x1E, 0xF1}, // İlk kartın dört byte'lık ID'si HEX olarak tanımlandı.
  {0xA6, 0x06, 0x1E, 0xF2}, // İkinci kart...
};

void setup() {
  servo.attach(9); // Servo motor 9 numaralı PWM çıkışına bağlı.
  Serial.begin(9600);                    //Bilgisayarın seri ekranıyla UART iletişimi.
  SPI.begin();                           // MFRC522 kart okuyucusuyla SPI iletişimi.
  kartOkuyucu.PCD_Init();                // MFRC522'yi başlat
  Serial.println("Lütfen kartı okutunuz.");
}

void loop() {
  if ( ! kartOkuyucu.PICC_IsNewCardPresent())  //Bir kart okunana kadar bekle.
    return;

  if ( ! kartOkuyucu.PICC_ReadCardSerial())  // Kartı oku.
    return;

  if (kartKontrol()) { // Kart geçerliyse yapılacaklar...
    Serial.println("Kart geçerli.");
    kapiAc(); // kapiAc fonksiyonunu çağır.
  }
  else { // Kart geçerli değilse yapılacaklar...
    for (i = 0; i < 4; i++) {
      Serial.print(kartOkuyucu.uid.uidByte[i], HEX); //Geçersiz kartın ID'sini ekrana yazdır.
```

```
    Serial.print(" ");
    }
    Serial.println(" numaralı kartınız sisteme tanımlı değil!");
  }
  delay(500);
  Serial.println("\nLütfen kartı okutunuz.");
}

boolean kartKontrol() { // Kart geçerliye 1, değilse 0 döndürür.
  k = 0;
  for (j = 0; j < kartSayisi; j++) { // Kayıtlı tüm kartlara bak.
    for (i = 0; i < 4; i++) { // Her kartın ID'si dört byte.
      if (kartlar[j][i] == kartOkuyucu.uid.uidByte[i]) {//Okunan kartı kayıtlı olanlarla karşı-
laştır.
        k++;
        if (k == 4) { // Tüm byte'lar doğrulandıysa...
          return 1; // 1 döndür.
        }
      }
    }
  }
  return 0; // Okunan kart kayıtlı kartlarla eşleşmediyse 0 döndür.
}

void kapiAc() {
  servo.write(90); //Servo motoru 90 dereceye getirir.
  delay(3000);
  servo.write(0); //Servo motoru 0 dereceye getirir.
  delay(1000);
}
```

2. Kart bilgilerini EEPROM'da tutan uygulamayı yapınız.

NOT Bu uygulamada ilk okutulan kart, hafızaya yönetici kartı olarak tanımlanır. Kart ekleme ve çıkarma işlemleri için önce yönetici kartı okutulur. Yönetici kartından sonra okutulan kart hafızada kayıtlıysa kart silinir. Hafızada kayıtlı değilse kart hafızaya eklenir.

```
#include <EEPROM.h>
#include <SPI.h> // Dahili SPI kütüphanesi
#include <MFRC522.h>  // v1.4.9

#define reset 9 // Reset pini
#define ss 10  // SS pin

MFRC522 kartOkuyucu(ss, reset);  // Kart okuyucu isimli nesne oluşturuldu.
```

```
byte i, j, k; /*j değişkeni kayıtlı kartların dört byte›lık ID›sinin ilk byte numarasını tutmaktadır. */
byte kartSayisi; //EEPROM hafızanın 0. baytında tutulacak.

void setup() {
  Serial.begin(9600);                    //Bilgisayarın seri ekranıyla UART iletişimi.
  SPI.begin();                           // MFRC522 kart okuyucusuyla SPI iletişimi.
  kartOkuyucu.PCD_Init();                // MFRC522'yi başlat.

  kartSayisi = EEPROM.read(0); // Cihaz yeniden başladığında kart sayısını EEPROM'dan al.

  if (!yoneticiKartKontrol()) { // Yönetici kartı sisteme tanımlı değilse...
    Serial.println("Lütfen yönetici olacak kartı okutunuz.");
    kartOku();
    for (i = 0; i < 4; i++) {
      EEPROM.write(1 + i, kartOkuyucu.uid.uidByte[i]);
    } // Yönetici kart ID'si EEPROM hafızanın 1. - 4. baytlarında tutulacak (İlk 5 bayt doldu.).
    Serial.println("Yönetici kartı hafızaya kaydedildi.");
  }
}
void loop() {
  kartOku();
  if (kartKontrol()) { // Kart geçerliyse yapılacaklar...
    Serial.println("Kart geçerli.");
  }
  else { // Kart geçerli değilse yapılacaklar...
    for (i = 0; i < 4; i++) {
      Serial.print(kartOkuyucu.uid.uidByte[i], HEX); //Geçersiz kartın ID'sini ekrana yazdır.
      Serial.print(" ");
    }
    Serial.println(" numaralı kartınız sisteme tanımlı değil!");
  }
  delay(500);
  Serial.println("\n Lütfen kartı okutunuz.");
}
void kartOku() {
  Serial.println("Lütfen kartı okutunuz.");
  if ( ! kartOkuyucu.PICC_IsNewCardPresent())  //Bir kart okunana kadar bekle.
    return;
  if ( ! kartOkuyucu.PICC_ReadCardSerial())  // Kartı oku.
    return;
}
boolean kartKontrol() { // Kart geçerliye 1, değilse 0 döndürür.
```

```
  if (yoneticiKartKontrol()) // Yönetici kartı okutulmuşsa...
    yonetimIslemleri(); // Kart ekleme ve silme işlemlerine geç.
  else { // Yönetici kartı değilse kart kontrol işlemi yap.
    k = 0;
    for (j = 5; j < EEPROM.length() / 4; j = +4) {
      /* Kayıtlı tüm kartlara bak. UNO hafızasına 1024/4=256 kart tanımlanabilir.
        j değişkeni kayıtlı kartların dört byte›lık ID›sinin ilk byte numarasını tutmaktadır.*/
      for (i = 0; i < 4; i++) { // Her kartın ID'si dört byte.
        if (EEPROM.read(j + i)  == kartOkuyucu.uid.uidByte[i]) { //Okunan kartı kayıtlı olanlar-
la karşılaştır. (5. byte'tan sonra)
          k++;
          if (k == 4) { // Tüm byte'lar doğrulandıysa...
            return 1; // 1 döndür.
          }
        }
      }
    }
    return 0; // Okunan kart kayıtlı kartlarla eşleşmediyse 0 döndür.
  }
}

boolean yoneticiKartKontrol() { // Yönetici kartıysa 1, değilse 0 döndürür.
  k = 0;
  for (i = 1; i < 5; i++) { /* 1. - 4. baytları kontrol et. (0. byteta kart sayısı tutulmaktadır.) */
    if (EEPROM.read(i) == kartOkuyucu.uid.uidByte[i]) { //Okunan kart yönetici kartı ise...
      k++;
      if (k == 4) { // Yönetici kartının tüm byteları doğrulandıysa...
        return 1; // 1 döndür.
      }
   }
  }
  return 0; // Okunan kart yönetici kartı değilse 0 döndür.
}

void yonetimIslemleri() {
  Serial.println("Lütfen eklemek veya silmek istediğiniz kartı okutunuz.");
  kartOku();
  if (kartKontrol)//Kart kayıtlıysa...
    kartSil(); // Kartı sil.
  else //Kart kayıtlı değilse...
    kartEkle(); //Kartı ekle.
}

void kartSil() {
```

```
  for (i = j; i < EEPROM.length() - j - 1; i++)
    EEPROM.write(i, EEPROM.read(i + 4));/*Bulunan kart ID'sinin ilk byte numarasından (j) başla-
yarak tüm hafızayı dört byte sola kaydır.*/
  Serial.println("Okutulan kart hafızadan silindi.");
}

void kartEkle() {
  for (i = 0; i < 4; i++)
    EEPROM.write(kartSayisi * 4 + 1 + i, kartOkuyucu.uid.uidByte[i]); /* Yeni kartı sıradaki
dört bytea yaz. */
  Serial.println("Yeni kart hafızaya kaydedildi.");
}
```

3. Geçerli kart okutulunca step motoru çalıştıran fonksiyonu yazınız.

```
void stepMotor() { // Step motor Yaklaşık 90° döner.
  int x, adimSayisi = 500, sure = 2;

  for (x = 0; x < adimSayisi / 4; x++) {
    for (byte i = 6; i <= 9; i++) { //Sağa dön.
      digitalWrite(i, 1);
      delay(sure);
      digitalWrite(i, 0);
    }
  }
  delay(1000); // Kapı açık kalma süresi.
  for (x = 0; x < adimSayisi / 4; x++) {
    for (byte i = 9; i >= 6; i--) { //Sola dön.
      digitalWrite(i, 1);
      delay(sure);
      digitalWrite(i, 0);
    }
  }
}
```

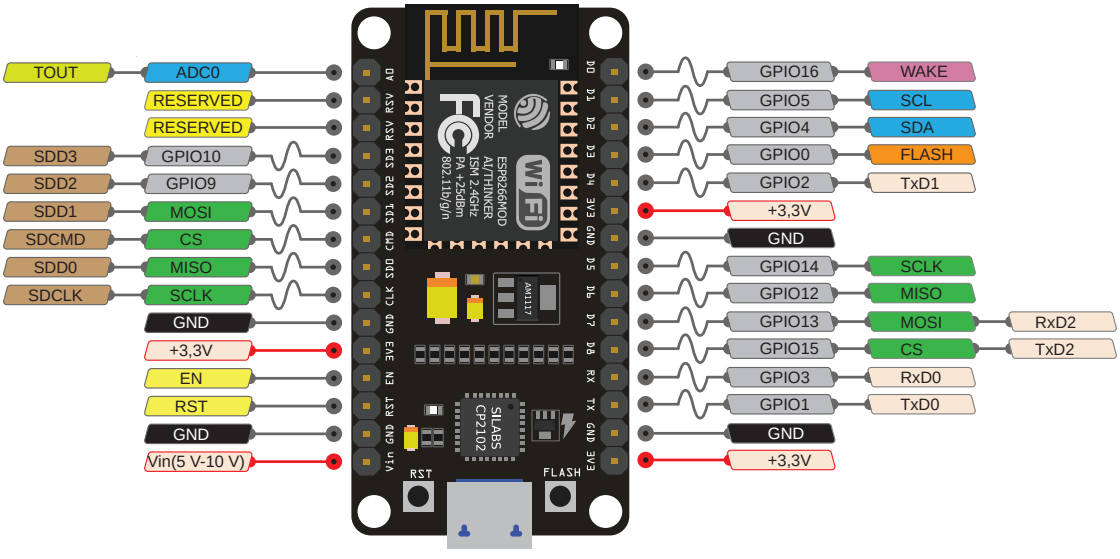
Sorular

1. Kart (verici) içindeki çip enerjisini nereden almaktadır? Belirtiniz.

2. Kart ID'leri string veri tipine dönüştürülüp kontrol edilebilir miydi? Açıklayınız.

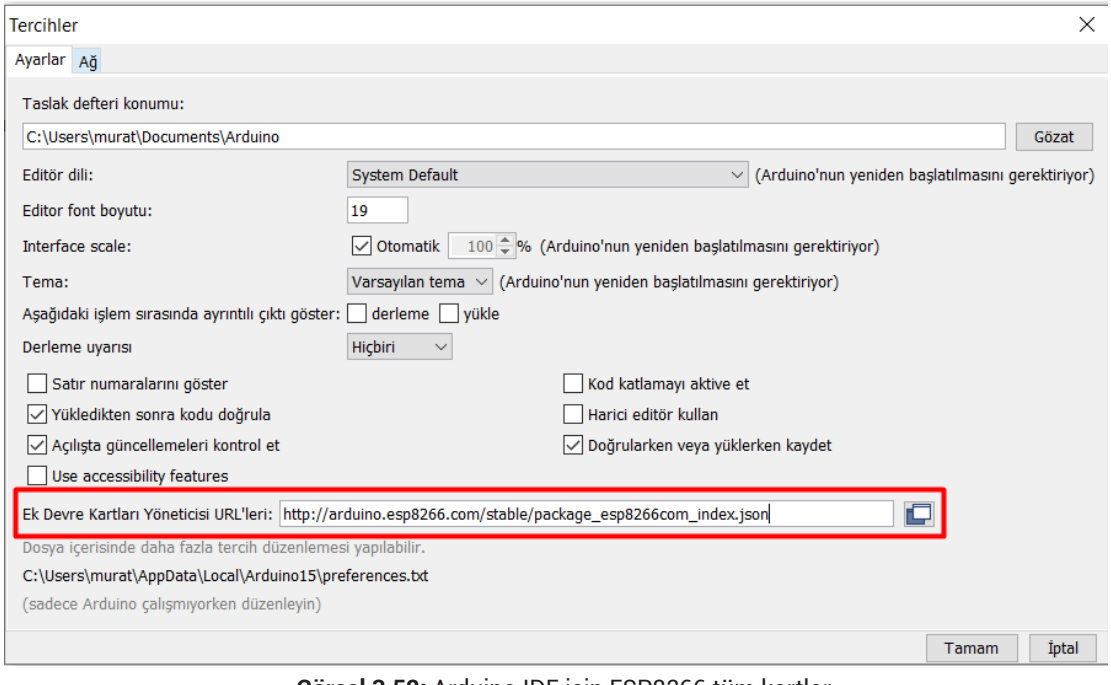
NodeMCU kartını Arduino IDE'ye tanıtma

Görsel 3.51'deki NodeMCU, ESP8266 **Wi-Fi** modülün uygulama kartı şeklidir. Bu kartın Arduino kartından farkı, maliyetinin düşük olması ve **Wi-Fi** modül içerdiğinden, IoT (Internet of Things - Nesnelerin İnterneti) kartı olarak kullanılmasıdır. ESP8266 mikrodenetleyicisini kullanan bu kartın işlemci frekansı 80 MHz, flash hafızası 4 MB ve RAM'i 80 KB'tır.



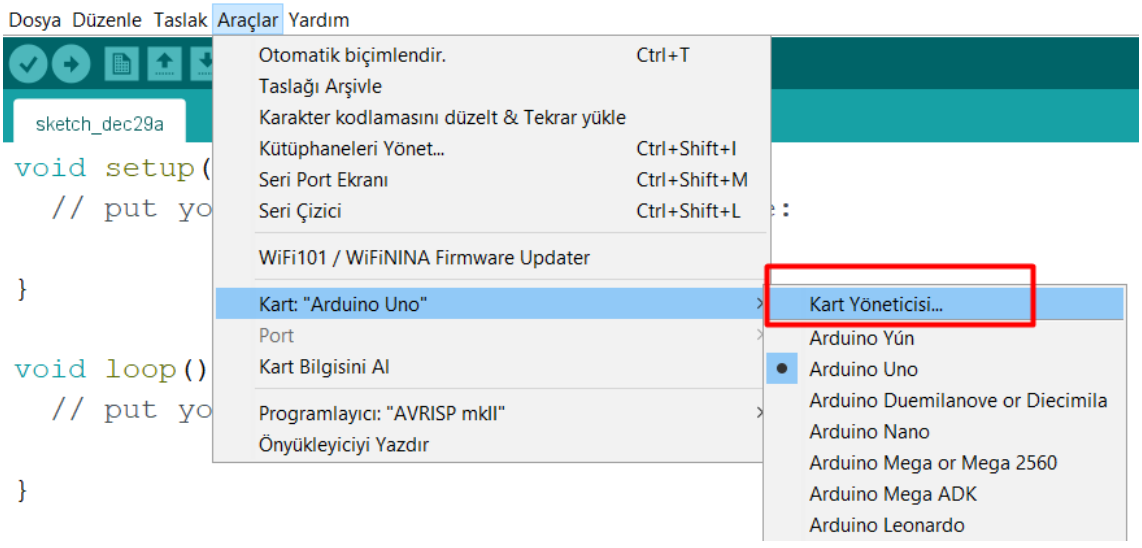
Görsel 3.51: NodeMCU pin yapısı

NodeMCU kartını Arduino IDE'ye tanıtmak için "Dosya → Tercihler" penceresindeki "Ek devre Kartları Yöneticisi URL'leri:" kutusuna http://arduino.esp8266.com/stable/package_esp8266com_index.json adresi yazılarak "Tamam" butonuna tıklanır (Görsel 3.52). Farklı kartları eklemek için (ESP 32 vb.) ilgili kartların bağlantı adresleri virgülle ayrılarak aynı kutuya girilir.



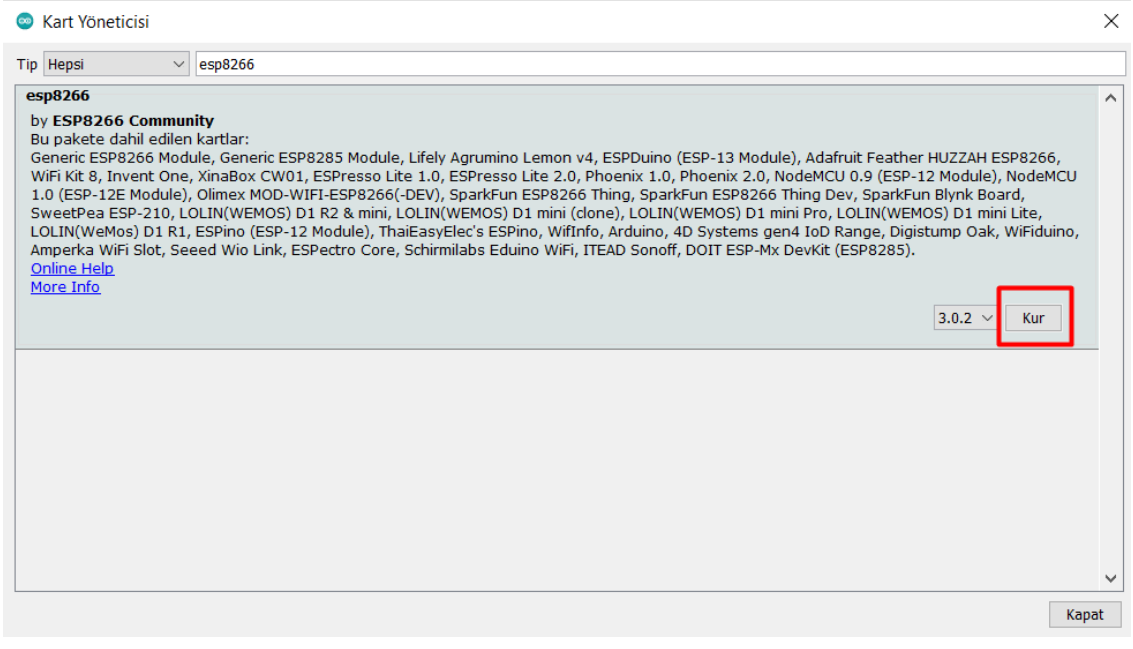
Görsel 3.52: Arduino IDE için ESP8266 tüm kartlar

"Araçlar → Kart: → Kart Yöneticisi" seçilerek kart yöneticisi penceresi açılır (Görsel 3.53).



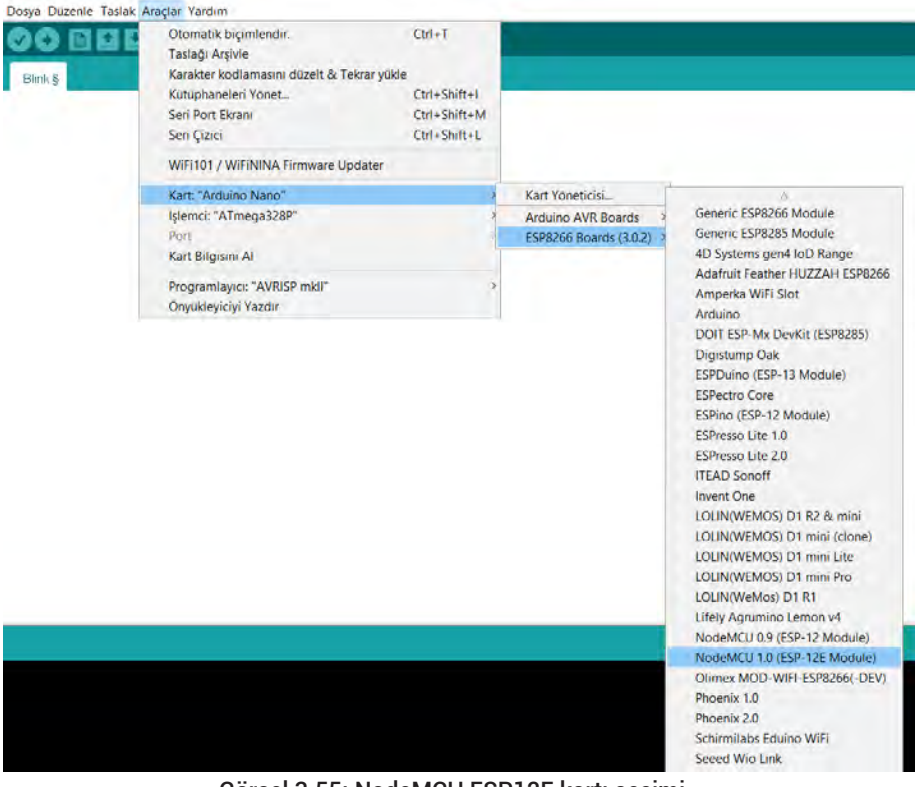
Görsel 3.53: Kart yöneticisi

Arama kutusuna "esp 8266" yazılır. "Kur" düğmesine tıklanarak versiyon 3.0.2 kurulur (Görsel 3.54).



Görsel 3.54: ESP8266 kartını Arduino IDE'ye tanıtma

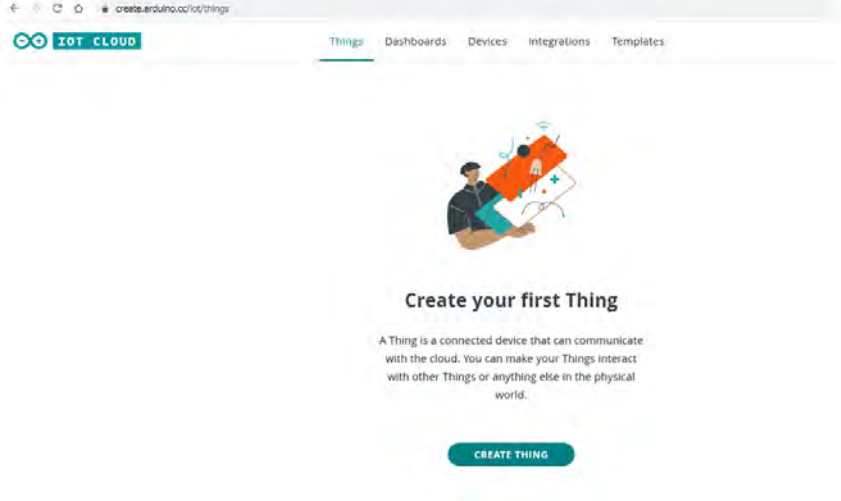
NodeMCU kartını seçmek için "Araçlar → Kart:→ ESP8266 Boards (3.0.2) → NodeMCU 1.0 (ESP-12E Module)" seçilir (Görsel 3.55). İlgili COM port seçilerek yazılan program karta yüklenir.



Görsel 3.55: NodeMCU ESP12E kartı seçimi

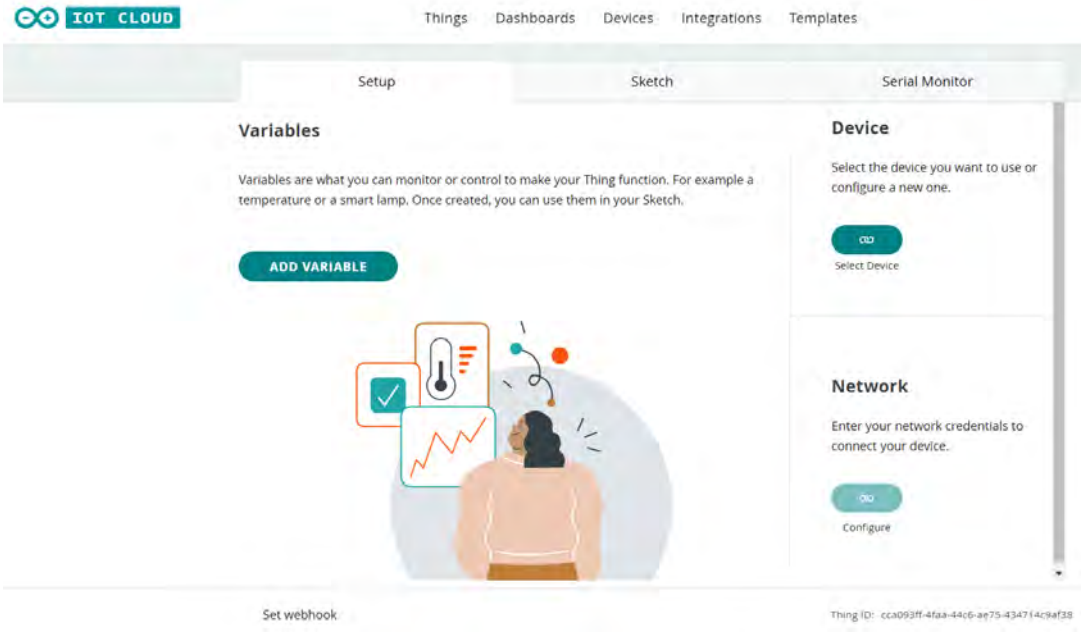
Nesnelerin İnterneti

Arduino IoT Cloud, Arduino tarafından geliştirilen IoT (internet of things) projeleri için tercih edilen bir bulut sistemidir. https://create.arduino.cc/iot adresinden ücretsiz olarak oluşturulabilen Arduino hesabıyla (Google vb. hesaplarla da giriş yapılabilir.) giriş yapılır. Görsel 3.56'daki "CREATE THING" butonuna basılarak yeni internet nesnesi oluşturulur.



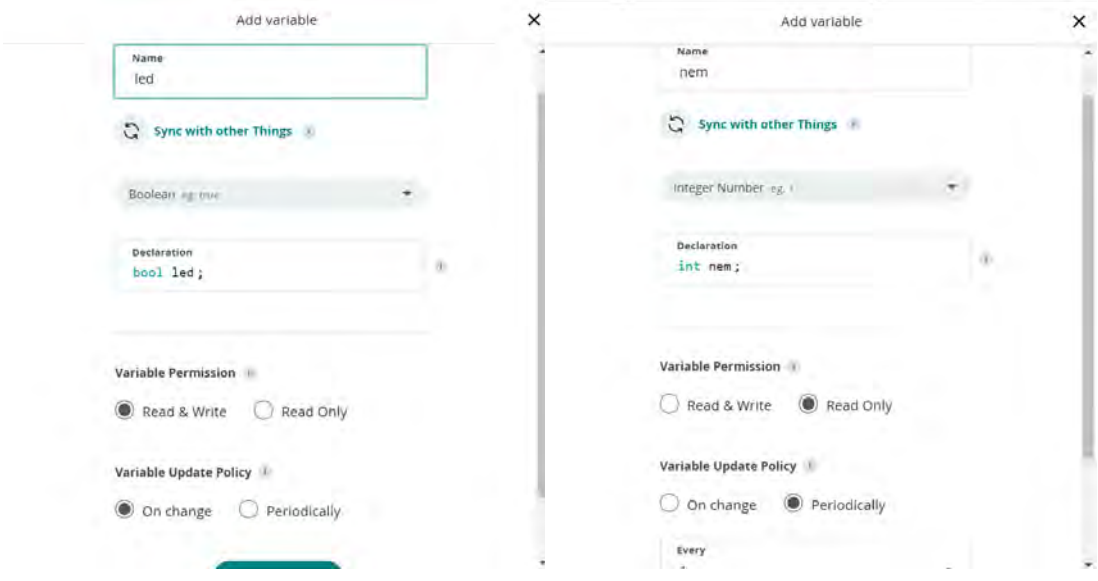
Görsel 3.56: Yeni internet nesnesi oluşturma

Görsel 3.57'de görülen ekrandan değişkenlerin, kullanılan mikrodenetleyici cihaz ve kablosuz ağın ayarları yapılır. "ADD VARIABLE" düğmesine tıklanarak değişkenler oluşturulur.



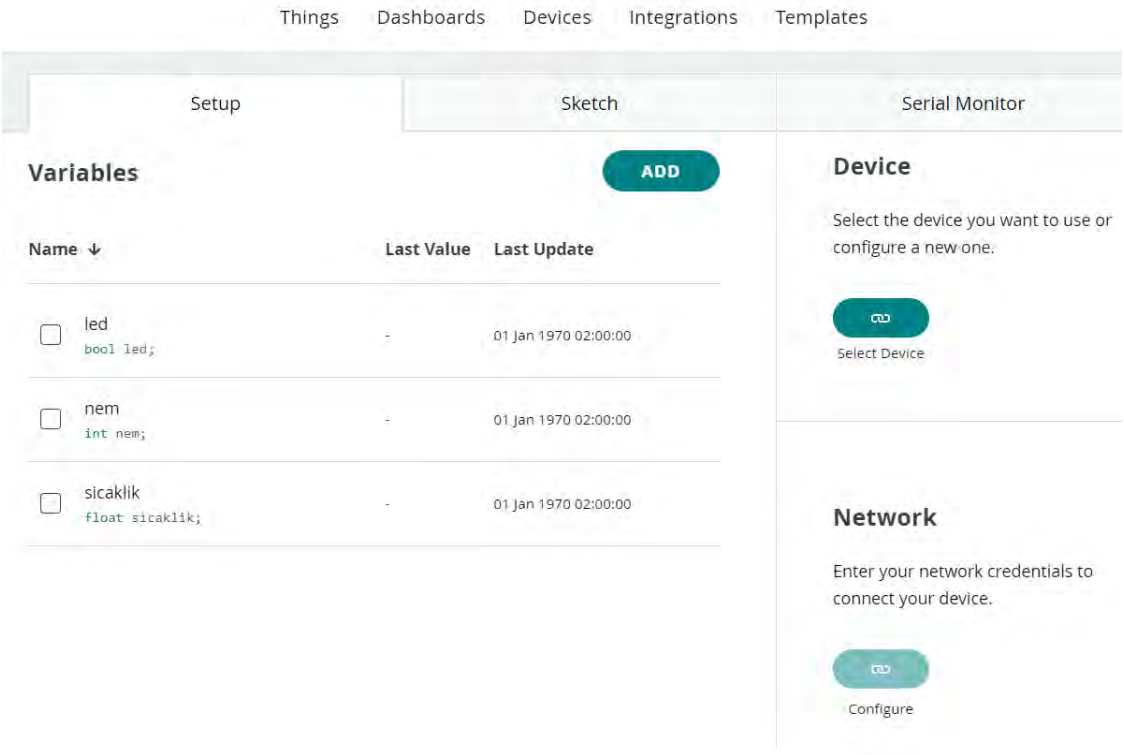
Görsel 3.57: Değişken ekleme

Görsel 3.58'de "led" isminde bool veri tipinde değişken oluşturulmuştur. Burada kullanılan değişken farklı bir nesnedeki aynı tip değişkenle senkronize de edilebilir. Sadece buluta veri gönderilecekse "Read Only", buluttan da mikrodenetleyiciye bilgi gidecekse "Read & Write" izni seçilir. Mikrodenetleyicide değişkenin içeriği değiştiğinde buluttaki bilginin güncellenmesi için "On change", değişkenin içeriği değişmese dahi seçilen süre sonunda buluttaki bilgilerin güncellenmesi için "Periodically" seçilir. "ADD VARIABLE" düğmesi tıklanarak değişken oluşturulur. "nem" ve "sicaklik" değişkenleri de benzer şekilde oluşturulur.

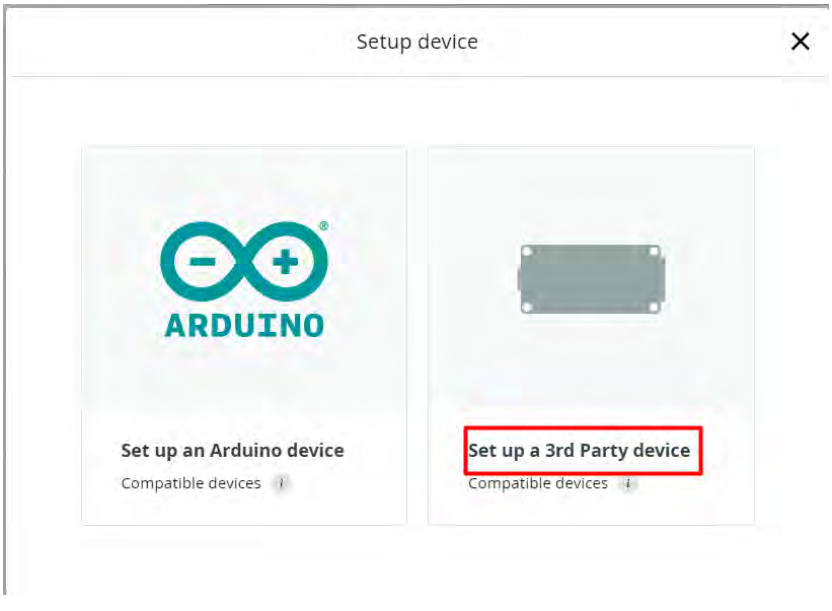


Görsel 3.58: Değişken özellikleri ayarlama

Görsel 3.59'daki "Select Devices" düğmesi tıklanarak Görsel 3.60'tan "Set up 3rd party devices" seçilir.

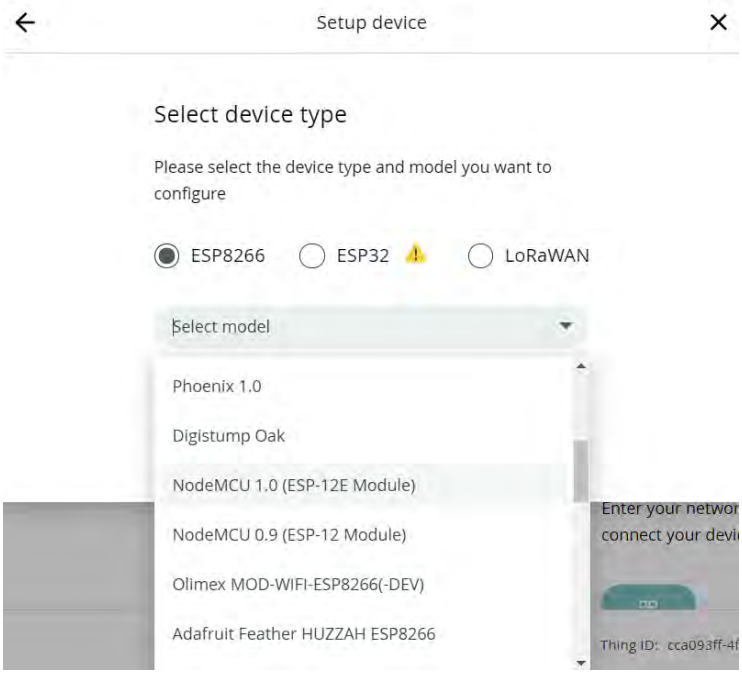


Görsel 3.59: Cihaz seçme



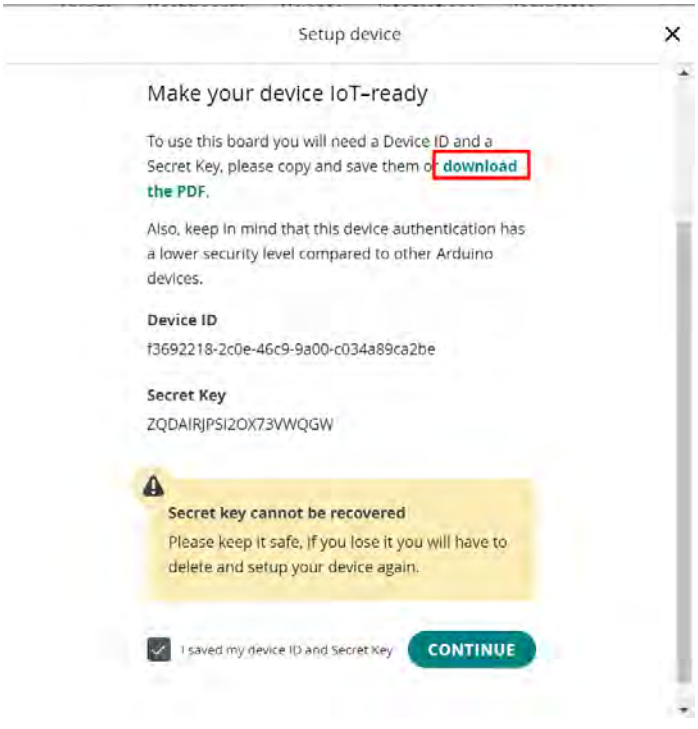
Görsel 3.60: Cihaz ayarları

Görsel 3.61'de görüldüğü gibi ESP8266 seçilerek listeden "NodeMCU 1.0 (ESP-12E Module)" tıklanır. Cihaza bir isim verilerek "next" düğmesine basılır.



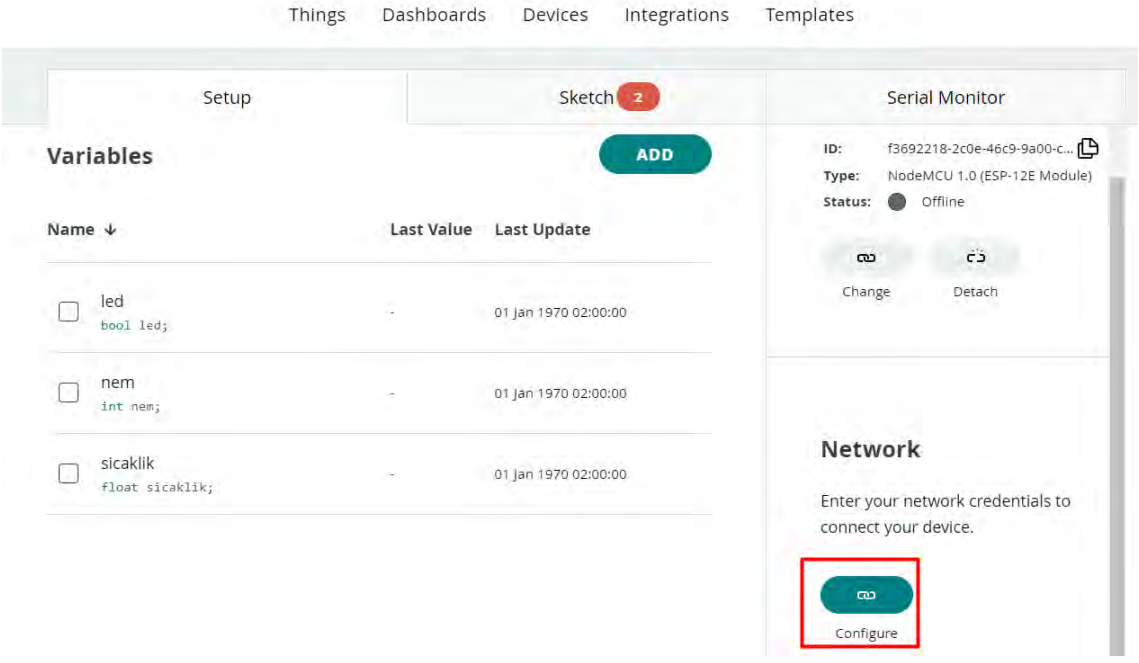
Görsel 3.61: Cihaz tipi seçme

Görsel 3.62'de "download" bağlantısı tıklanarak Device ID ve Secret Key PDF olarak kaydedilir. CONTINUE düğmesiyle cihaz ekleme işlemleri tamamlanır.



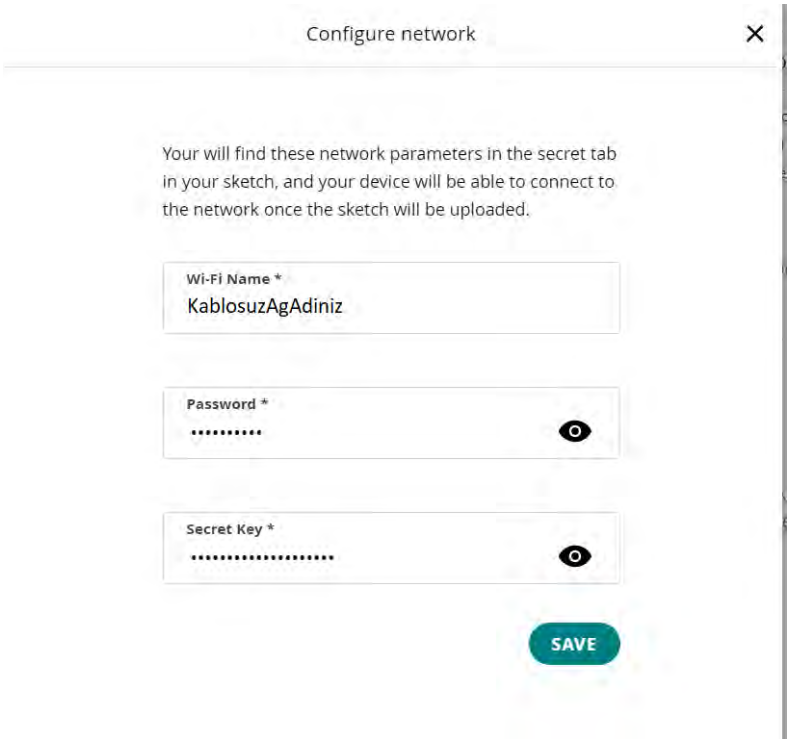
Görsel 3.62: Özel anahtar oluşturma

Wi-Fi ayarlarının yapılması için Görsel 3.63'te görüldüğü gibi Network kısmından Configure düğmesine tıklanır.



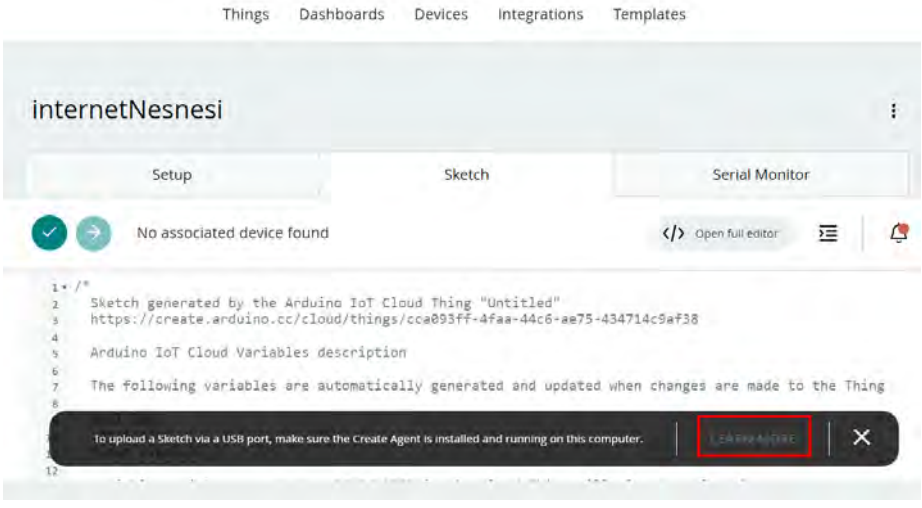
Görsel 3.63: Kablosuz ağ ayarları

Cihazın bağlantı kuracağı kablosuz internetin kullanıcı adı, şifresi ve Görsel 3.62'deki Secret Key girilerek SAVE düğmesi tıklanır (Görsel 3.64).



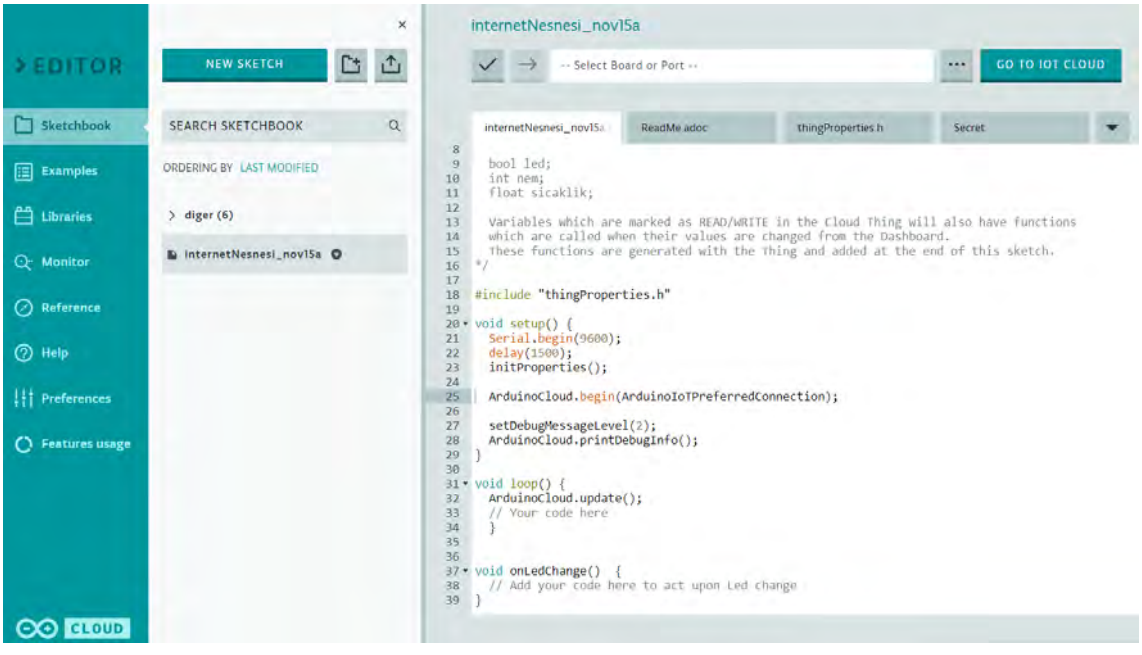
Görsel 3.64: Kablosuz ağ bilgileri

Görsel 3.65'te görüldüğü gibi Sketch sekmesine tıklanır. Cihazı Arduino web editörü üzerinden programlayabilmek için Create Agent eklentisi yüklü olmalıdır. LEARN MORE düğmesine tıklanarak açılan pencereden download bağlantısı tıklanır, Create Agent eklentisi bilgisayara indilir ve kurulur. Create Agent eklentisi bilgisayarda arka planda sürekli çalışır. Open full editör düğmesi tıklanarak Arduino web editörü açılır.



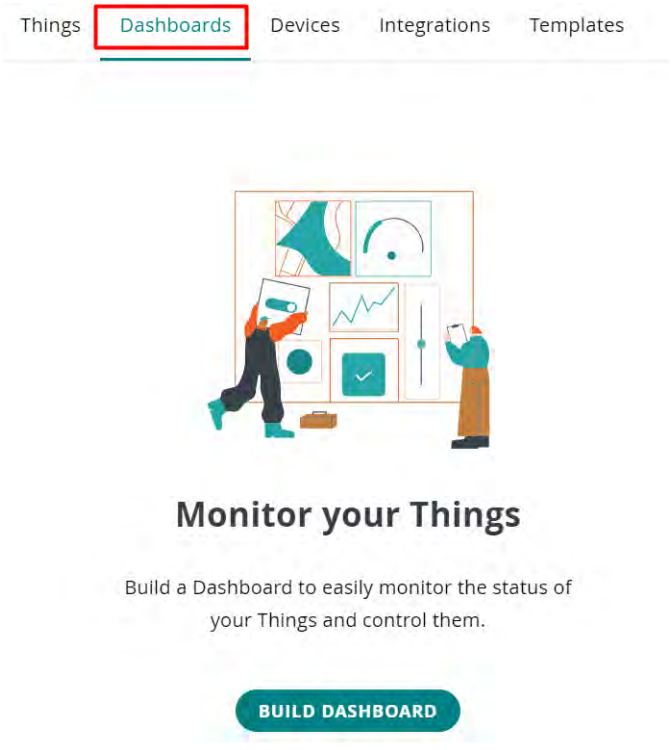
Görsel 3.65: Arduino Create Agent eklentisi yükleme

Görsel 3.66'da Arduino web editörde internet nesnesine ait kodlar hazır gelmektedir. Kullanılacak değişkenler önceden tanımlandığından (thingProperties.h sekmesinde değişkenlerin tanımlandığı görülebilir.) tekrar tanımlamaya gerek yoktur. Değişkenler kullanılarak program loop() fonksiyonu içine yazılır. LED'i yakma butonu read/write olarak tanımlandığından (buluttan cihaza giden bilgi olduğundan) LED'i yakma butonuna ait komutlar void onLedChange() fonksiyonu içine yazılır.



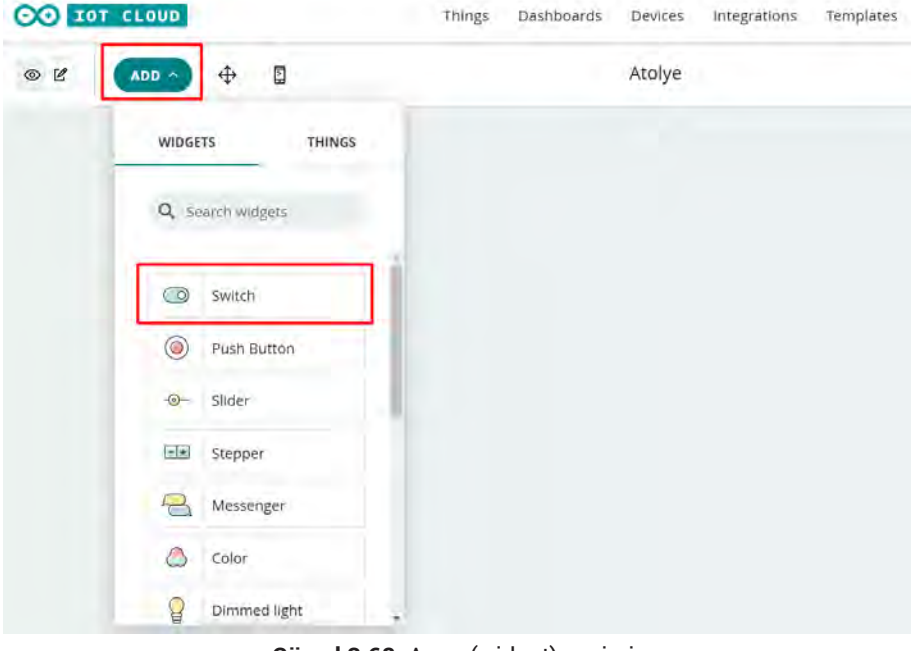
Görsel 3.66: Arduino web editörü

Görsel 3.67'de gösterge panelini oluşturmak için Dashboard sekmesinden "BUILD DASHBOARD" düğmesine tıklanır.



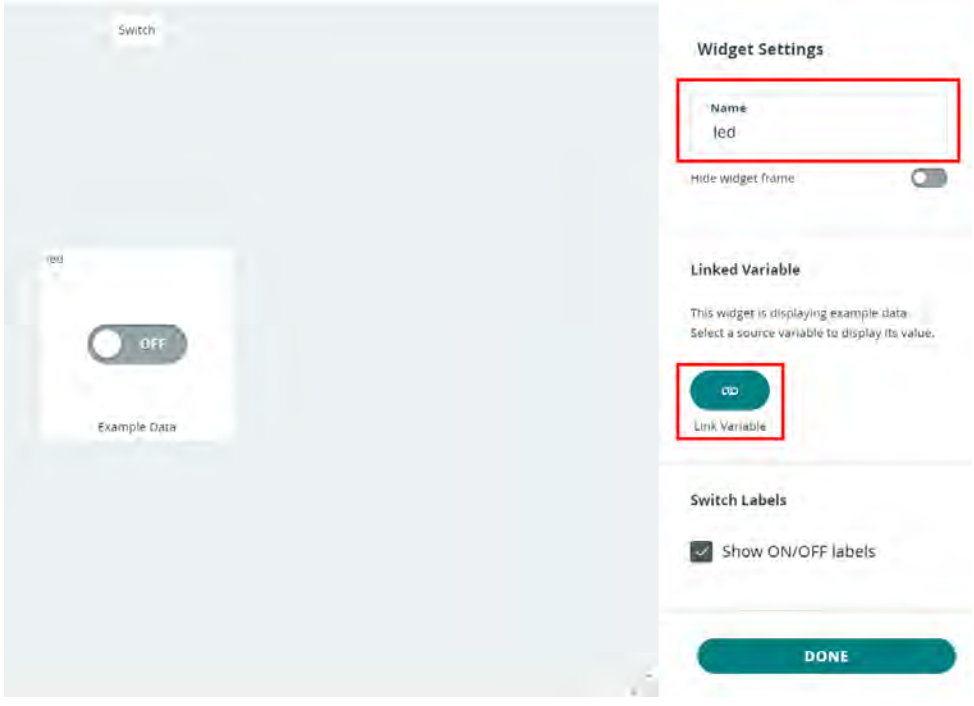
Görsel 3.67: Gösterge paneli oluşturmak

Görsel 3.68'de "ADD" düğmesine tıklanarak switch aracı (widget) seçilir.



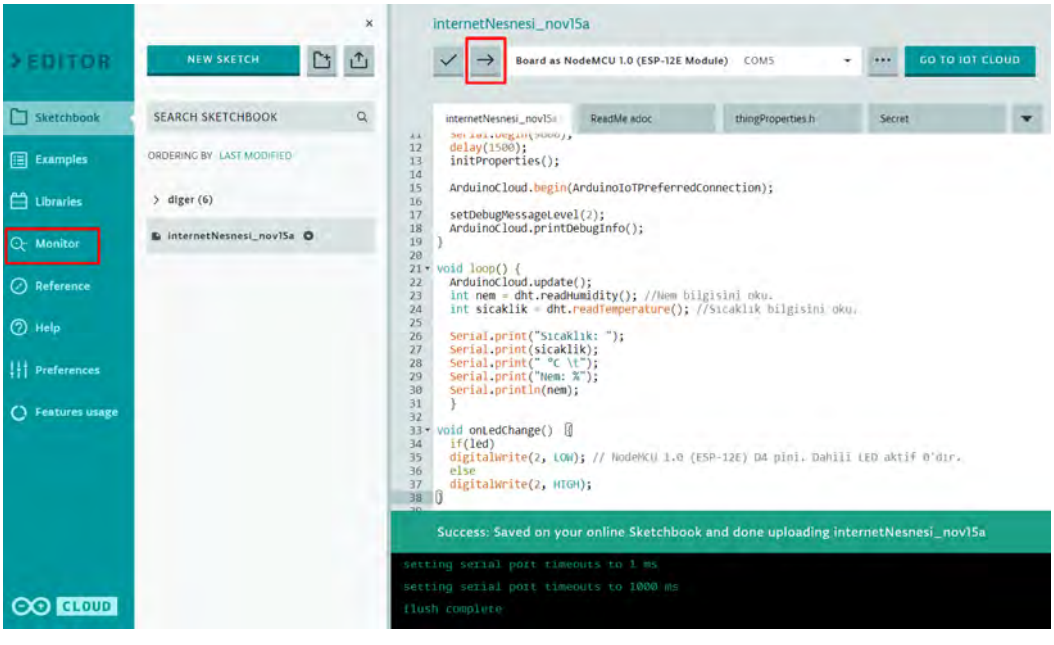
Görsel 3.68: Aracı (widget) seçimi

Görsel 3.69'da görüldüğü gibi switch aracına isim verilebilir. Link Variable düğmesine tıklanarak açılan pencereden önceden tanımlı "led" değişkeni seçilir. "DONE" düğmesiyle işlem tamamlanır. Sıcaklık için **gauge**, nem için **percantage** veya **chart** araçları da benzer şekilde gösterge paneline eklenir.



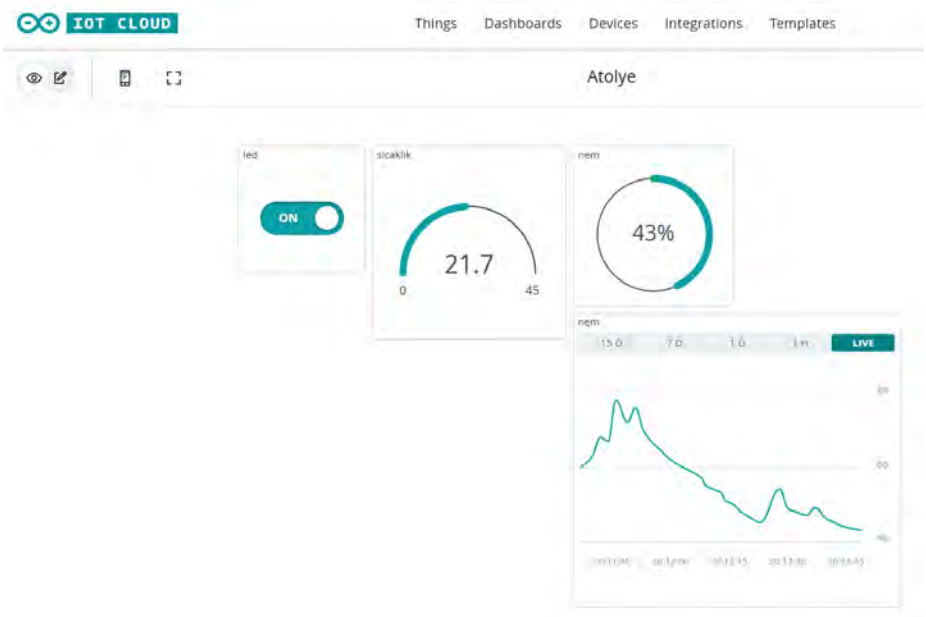
Görsel 3.69: Switch aracına isim verme

Görsel 3.70'te görüldüğü gibi yükleme düğmesiyle program cihaza yüklenir. Monitör kısmından bağlantı çıktıları, sıcaklık ve nem değerleri de ayrıca gözlemlenebilir.



Görsel 3.70: Programı cihaza yükleme

Görsel 3.71'de görüldüğü gibi Dasboards sekmesinden nem ve sıcaklık bilgileri gözlemlenir ve "led" isimli switch aracıyla NodeMCU üzerindeki dâhilî LED yakılıp söndürülür.



Görsel 3.71: Gösterge panelini kullanma

Görsel 3.72'de verildiği gibi gösterge paneline akıllı telefondan https://create.arduino.cc/iot/dashboards/ adresi üzerinden ulaşılarak da aynı işlemler gerçekleştirilir.

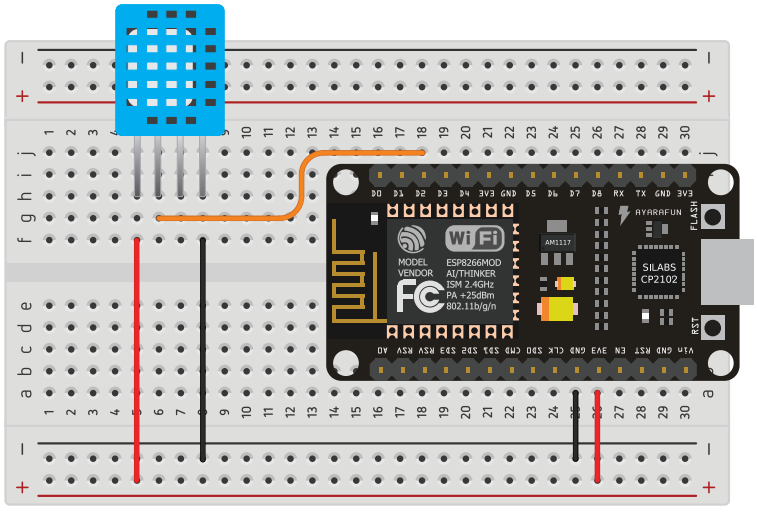


Görsel 3.72: Gösterge paneline akıllı telefondan erişim

Uygulama Adı: Wi-Fi Uygulaması — No.: 14

Amaç: Wi-Fi uygulaması yapmak.

Bu uygulamada NodeMCU 1.0 (ESP-12E) kullanılarak Arduino IoT Cloud üzerinden başka bir konumdaki LED'i yakıp söndürme ve ortamın nem ve sıcaklık bilgisini bilgisayar veya akıllı telefon üzerinden görüntüleme yapılmaktadır (Görsel 3.73).



Görsel 3.73: Gösterge paneline akıllı telefondan erişim

Wi-Fi uygulaması programı aşağıdaki gibidir:

```
#include "thingProperties.h"
#include "DHT.h" //Adafruit DHT Sensor Library v1.2.3

#define DHTpin 4     // Sensörün bağlandığı pin. NodeMCU 1.0 (ESP-12E) D2 pini.
#define DHTtipi DHT11   // Sensör tipi: DHT11, DHT21, DHT22.

DHT dht(DHTpin, DHTtipi); //dht isimli nesne oluşturuldu.

void setup() {
  pinMode(2, OUTPUT);     //  GPIO2 pini D4 pini çıkış. (Dahili LED.)
  pinMode(4, INPUT);     //  GPIO4 pini D2 pini giriş.
  dht.begin();
  Serial.begin(9600);
  delay(1500);
  initProperties();

  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}
```

```
void loop() {
  ArduinoCloud.update();
  nem = dht.readHumidity(); //Nem bilgisini oku.
  sicaklik = dht.readTemperature(); //Sıcaklık bilgisini oku.
  Serial.print("Sıcaklık: ");
  Serial.print(sicaklik);
  Serial.print(" °C \t");
  Serial.print("Nem: %");
  Serial.println(nem);
}
void onLedChange()  {
  if (led)
  digitalWrite(2, LOW); // NodeMCU 1.0 (ESP-12E) D4 pini. Dahili LED aktif 0'dır.
  else
  digitalWrite(2, HIGH);
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.73'teki devreyi kurunuz.
4. https://create.arduino.cc/iot adresinden internet nesnesini oluşturunuz.
5. Arduino web editöründen (https://create.arduino.cc/editor/) nodeMCU kartını programlayınız.
6. https://create.arduino.cc/iot/dashboards/ adresinden göstergeleri izleyerek LED'i internet üzerinden kontrol ediniz.
7. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Röle modülü kullanarak 220 V lamba veya motor kontrol eden uygulamayı gerçekleştiriniz.

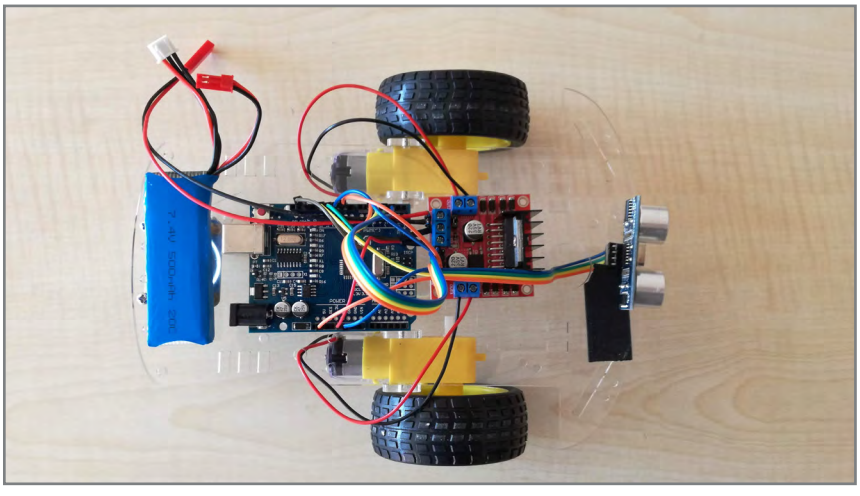
Sorular

1. Yapılan bu uygulamada Arduino web editöre neden ihtiyaç duyulmaktadır? Belirtiniz.
2. Arduino IOT uygulaması için google firebase ve php web server kullanımını araştırınız.

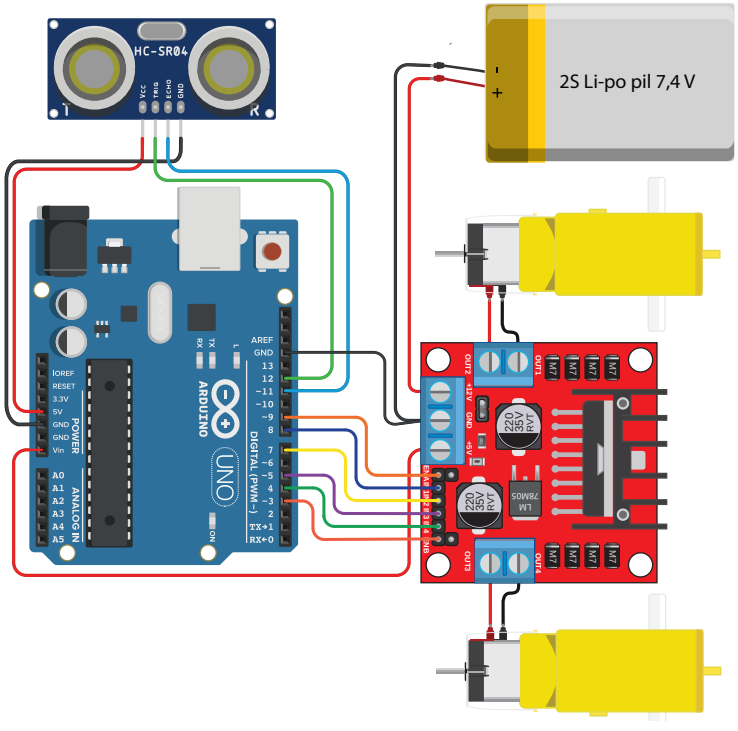
Uygulama Adı: Engelden Kaçan Robot Uygulaması No.: 15

Amaç: Engelden kaçan robot uygulaması yapmak.

Engelden kaçan robotta engelleri algılamak için ultrasonik sensör kullanılmaktadır (Görsel 3.74). Ultrasonik sensörden gelen bilgilerle mesafe hesaplanır. Mesafe 30 cm'nin altına indiğinde motor tarafından sürücüye sol tekerleği geriye döndürme bilgisi gönderilmektedir. 30 cm üzerindeki tüm mesafe ölçümlerinde tekerlekler ileri yönde döndürülecek şekilde motor sürücüye bilgi gönderilmektedir. Taslaktaki "hiz=255;" değeri düz gitme ve dönüş için farklı değerlere ayarlanarak robot yavaşlatılabilir. Görsel 3.75'te engelden kaçan robot uygulama devresi görülmektedir.



Görsel 3.74: Engelden kaçan robot



Görsel 3.75: Engelden kaçan robot uygulaması

Engelden kaçan robot uygulaması programı aşağıdaki gibidir:

```
const byte IN1 = 8, IN2 = 7, IN3 = 5, IN4 = 4; // Motor sürücü yön pinleri
const byte enA = 9, enB = 3; // Motor sürücü PWM pinleri
const byte echo = 11, trig = 12; // Ultrasonik sensör pinleri

byte hiz = 255; // PWM.
unsigned long sure;
int uzaklik;

void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
}

void loop() {
  digitalWrite(trig, 0); // Darbe sinyali gönder.
  delayMicroseconds(10);
  digitalWrite(trig, 1);
  delayMicroseconds(10);

  sure = pulseInLong(echo, 1); // Gelen sinyalin HIGH'da kalma süresini oku.
  uzaklik = (0.0343 * sure) / 2; // Mesafeyi hesapla.

  if (uzaklik < 30) { // Uzaklık 30 cm'den yakınsa sola dön.
    digitalWrite(IN1, 0);
    digitalWrite(IN2, 1);
    analogWrite(enA, hiz);
    analogWrite(enB, hiz);
  }
  else { // Uzaklık 30 cm'den yakın değilse düz git.
    digitalWrite(IN1, 1);
    digitalWrite(IN2, 0);
    digitalWrite(IN3, 1);
    digitalWrite(IN4, 0);
    analogWrite(enA, hiz);
    analogWrite(enB, hiz);
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.75'teki devreyi kurunuz. Programı yükleyiniz.
4. Robotu zemine koyarak robotun çalışmasını gözlemleyiniz.
5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Motor sürücünün yön girişlerinde (IN1, IN2, IN3, IN4) analogWrite() fonksiyonunu kullanarak devreyi yeniden düzenleyiniz (ENA ve ENB **jumperl**arı takılı kalır.).

```
const byte IN1 = 10, IN2 = 9, IN3 = 6, IN4 = 5; // PWM pinleri. (enA ve enB'de jumper takılı).
const byte echo = 11, trig = 12; // Ultrasonik sensör pinleri

byte hiz = 70; // PWM.
unsigned long sure;
int uzaklik;

void setup() {
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
}

void loop() {
  digitalWrite(trig, 0); // Darbe sinyali gönder.
  delayMicroseconds(10);
  digitalWrite(trig, 1);
  delayMicroseconds(10);

  sure = pulseInLong(echo, 1); // Gelen sinyalin HIGH'da kalma süresini oku.
  uzaklik = (0.0343 * sure) / 2; // Mesafeyi hesapla.

  if (uzaklik < 30) { // Uzaklık 30 cm'den yakınsa sola dön.
    analogWrite(IN1, 0);
    analogWrite(IN2, hiz);
    analogWrite(IN3, hiz);
    analogWrite(IN4, 0);
  }
  else { // Uzaklık 30 cm'den yakın değilse düz git.
    analogWrite(IN1, hiz);
    analogWrite(IN2, 0);
    analogWrite(IN3, hiz);
    analogWrite(IN4, 0);
```

```
  }
}
```

2. Üç adet ultrasonik sensörü ön tarafa 15° lik açılarla yerleştirerek uygulamayı gerçekleştiriniz.

```
const byte IN1 = 10, IN2 = 9, IN3 = 6, IN4 = 5; // PWM pinleri. (enA ve enB'de jumper takılı).
const byte echoSag = 3, trigSag = 4; // Sağ ultrasonik sensör pinleri
const byte echoOrta = 7, trigOrta = 8; // Orta ultrasonik sensör pinleri
const byte echoSol = 12, trigSol = 13; // Sol ultrasonik sensör pinleri

byte hiz = 100; // PWM.
unsigned long sure;
int uzaklik;

void setup() {
  pinMode(trigSag, OUTPUT);
  pinMode(echoSag, INPUT);
  pinMode(trigOrta, OUTPUT);
  pinMode(echoOrta, INPUT);
  pinMode(trigSol, OUTPUT);
  pinMode(echoSol, INPUT);
}

void loop() {

  byte uzaklikSag = mesafe(trigSag, echoSag);
  byte uzaklikOrta = mesafe(trigOrta, echoOrta);
  byte uzaklikSol = mesafe(trigSol, echoSol);

  if (uzaklikSag < 50)
    sag();

  if (uzaklikSol < 50)
    sol();


  if (uzaklikOrta < 30) { // Uzaklık 30 cm'den yakınsa dur, geri git, sola dön.
    dur();
    delay(1000);
    geri();
    delay(300);
    sol();
    delay(400);
  }
  else  ileri(); // Uzaklık 30 cm'den yakın değilse düz git.
}
byte mesafe(byte trig, byte echo) {
```

```
  digitalWrite(trig, 0); // Darbe sinyali gönder.
  delayMicroseconds(10);
  digitalWrite(trig, 1);
  delayMicroseconds(10);

  sure = pulseInLong(echo, 1); // Gelen sinyalin HIGH'da kalma süresini oku.
 uzaklik = (0.0343 * sure) / 2; // Mesafeyi hesapla.
  return uzaklik;
}
void ileri() {
  analogWrite(IN1, hiz);
  analogWrite(IN2, 0);
  analogWrite(IN3, hiz);
  analogWrite(IN4, 0);
}
void geri() {
  analogWrite(IN1, 0);
  analogWrite(IN2, hiz);
  analogWrite(IN3, 0);
  analogWrite(IN4, hiz);
}
void sag() {
  analogWrite(IN1, hiz);
  analogWrite(IN2, 0);
  analogWrite(IN3, 0);
  analogWrite(IN4, 0);
}
void sol() {
  analogWrite(IN1, 0);
  analogWrite(IN2, 0);
  analogWrite(IN3, hiz);
  analogWrite(IN4, 0);
}
void dur() {
  analogWrite(IN1,  0);
  analogWrite(IN2, 0);
  analogWrite(IN3, 0);
  analogWrite(IN4, 0);
}
```

Sorular

1. İleri ve dönüş hızları farklı ayarlanabilir mi? Belirtiniz.
2. Motor sürücünün yön girişlerinde analogWrite() fonskiyonu kullanıldığında IN1, IN2, IN3, IN4 uçlarını çıkış olarak ayarlamak gerekir mi? Nedenini açıklayınız.

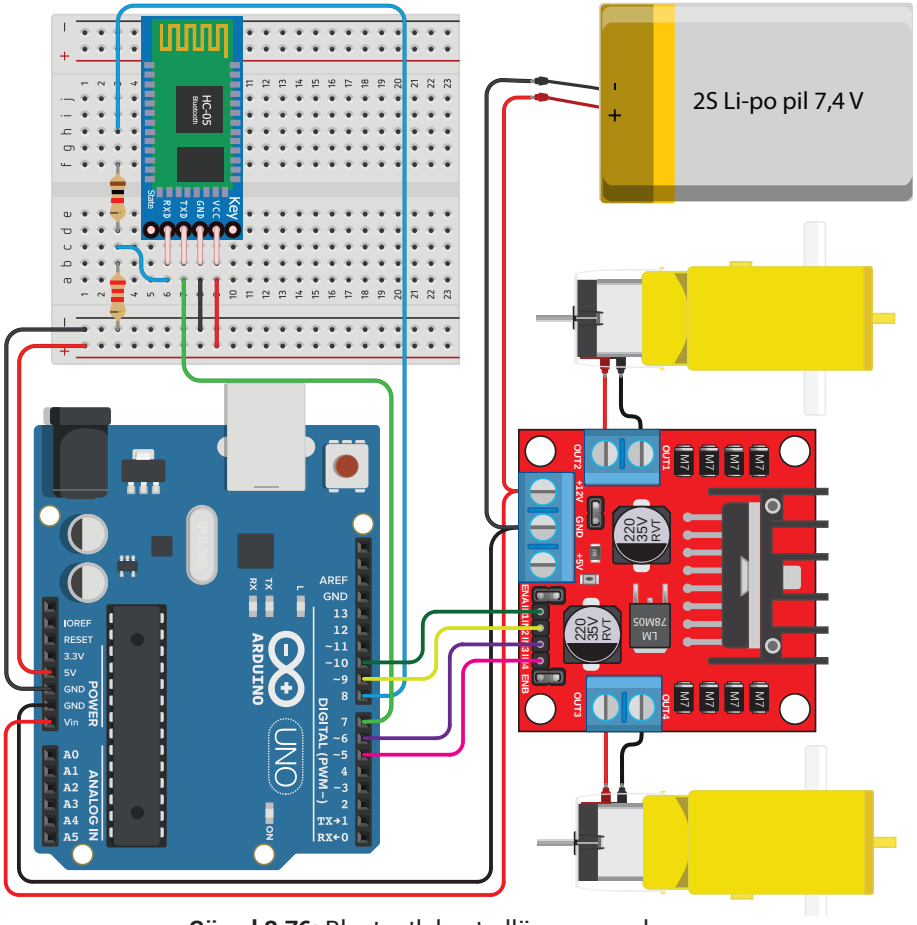
Uygulama Adı: Bluetooth Kontrollü Araç Uygulaması No.: 16

Amaç: Bluetooth kontrollü araç uygulaması yapmak.

Görsel 3.76'da bluetooth modülün RX ve TX pinlerinin SoftwareSerial.h kütüphanesi kullanılarak 8 ve 7 numaralı pinlere bağlanması gösterilmiştir. Modülün TX ucu Arduino'nun RX (8. pin) ucuna, modülün RX ucu Arduino'nun TX (7. pin) ucuna bağlanmalıdır. Ancak modülün TX ucundan çıkan 3,3 V Arduino RX ucunda lojik 1 olarak kabul edilirken Arduino'nun TX ucundan çıkan 5 V modülün RX ucuna doğrudan uygulandığında bozulmasına neden olacaktır. Bu nedenle Arduino'nun TX ucundan çıkan 5 V'u 3,3 V'a düşürmek için gerilim bölücü dirençler kullanılmalıdır.

Motor sürücüde ENA ve ENB uçlarındaki **jumper** takılı bırakılarak diğer dört girişten analogWrite fonksiyonuyla PWM verilmiştir.

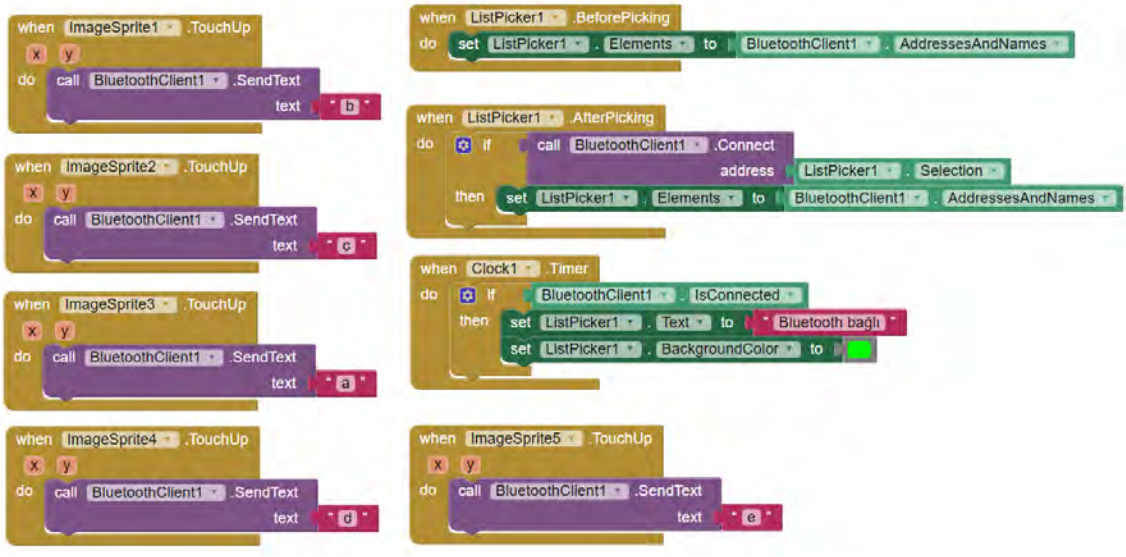
Devreye enerji verildiğinde bluetooth modülle haberleşmek için Android telefondan bluetooth bağlantısı açılarak hc-05 cihazı bulunur. Varsayılan şifre "1234" veya "0000"dır. Android telefondan veri gönderebilmek için **play store**dan "Arduino bluetooth controller" benzeri uygulamalar kullanılır (Görsel 3.77) veya MIT app invertor ile akıllı telefon uygulaması yapılır (Görsel 3.78). Telefondan gönderilen karakterlere göre aracın yönü belirlenir.



Görsel 3.76: Bluetooth kontrollü araç uygulaması



Görsel 3.77: Bluetooth araç Android uygulaması



Görsel 3.78: Akıllı telefon uygulaması MIT app invertor kodları

Bluetooth kontrollü araç uygulaması programı aşağıdaki gibidir:

```
#include <SoftwareSerial.h> // Arduino IDE ile gelir. Kurulum gerektirmez.
SoftwareSerial bluetooth(7, 8); // RX, TX (Arduino tarafında).
const byte IN1 = 10, IN2 = 9, IN3 = 6, IN4 = 5; // PWM pinleri. (enA ve enB'de jumper takılı).
char karakter;
byte hiz = 255; // Başlangıç hızı.

void setup() {
  bluetooth.begin(9600); // Bluetooth iletişimi başlat.
}
void loop() {
  if (bluetooth.available()) { // Veri geliyorsa...
    karakter = bluetooth.read(); // Gelen karakteri karakter değişkenine ata.
    if (karakter == 'a') { // İleri.
      analogWrite(IN1, hiz);
      analogWrite(IN2, 0);
```

```
      analogWrite(IN3, hiz);
      analogWrite(IN4, 0);
    } else if (karakter == 'b') { // Geri.
      analogWrite(IN1, 0);
      analogWrite(IN2, hiz);
      analogWrite(IN3, 0);
      analogWrite(IN4, hiz);
    } else if (karakter == 'c') { // Sol.
      analogWrite(IN1, hiz);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, hiz);
    } else if (karakter == 'd') { // Sağ.
      analogWrite(IN1, 0);
      analogWrite(IN2, hiz);
      analogWrite(IN3, hiz);
      analogWrite(IN4, 0);
    } else if (karakter == 'e') { // Dur.
      analogWrite(IN1, 0);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, 0);
    }
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.76'daki devreyi kurunuz. Programı yükleyiniz.
4. Telefondan bluetooth açarak modülle eşleştiriniz.
5. Telefondan uygulamayla a, b, c, d, e karakterleri göndererek aracı yönlendiriniz.
6. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

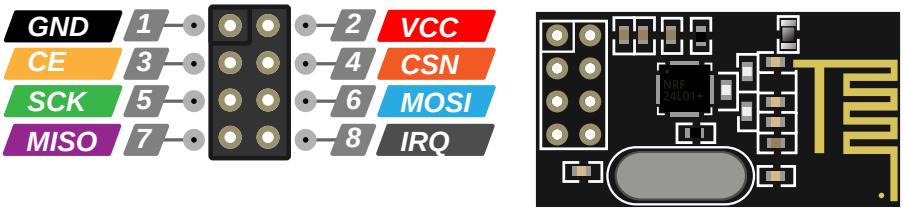
1. Bluetooth araçta, telefondan **"k"** harfi gönderildiğinde korna çalan (**buzzer**), **"f"** harfi gönderildiğinde farları yakan (LED) uygulamayı gerçekleştiriniz.

Soru

1. **UART** iletişim için 0 ve 1 numaralı pinler neden tercih edilmemiştir? Açıklayınız.

Kablosuz Haberleşme

Görsel 3.79'da verilen nRF24L01+ alıcı-verici modülü, 2,4 GHz ISM frekans bandında çalışmak üzere tasarlanmıştır. Modül, veri iletimi için GFSK modülasyonunu kullanır. Veri aktarım hızı 250 kbps, 1 Mbps ve 2 Mbps olabilir. nRF24L01+ alıcı-verici modülü, maksimum 10 Mbps veri hızıyla dört pinli Seri Çevresel Arabirim (SPI) üzerinden iletişim kurar. Antensiz modülün mesafesi 100 m iken antenli modül 1000 m mesafeden iletişim kurabilir.



Görsel 3.79: nRF24L01+ pin yapısı

VCC: Topraklama pini.

GND: 3,3 V besleme. Bu modüle uygun 5 V adaptör kullanıldığında 5 V verilebilir. Adaptör kullanılmadan 5 V verilirse modül bozulur.

CE (Chip Enable): Lojik 1 yapıldığında nRF24L01, o anda hangi modda olduğuna bağlı olarak ya gönderir ya da alır.

CSN (Chip Select Not): Lojik 0 yapıldığında nRF24L01 veri için SPI portunu dinlemeye başlar.

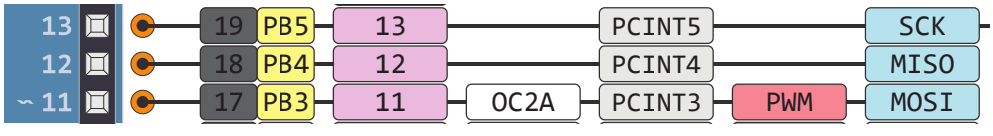
SCK (Serial Clock): SPI bus **master** tarafından sağlanan saat darbelerini kabul eder.

MOSI (Master Out Slave In): nRF24L01'e SPI girişidir.

MISO (Master In Slave Out): nRF24L01'den SPI çıkışıdır.

IRQ: İşlenecek yeni veriler olduğunda **master** cihazı uyaran bir kesme pinidir.

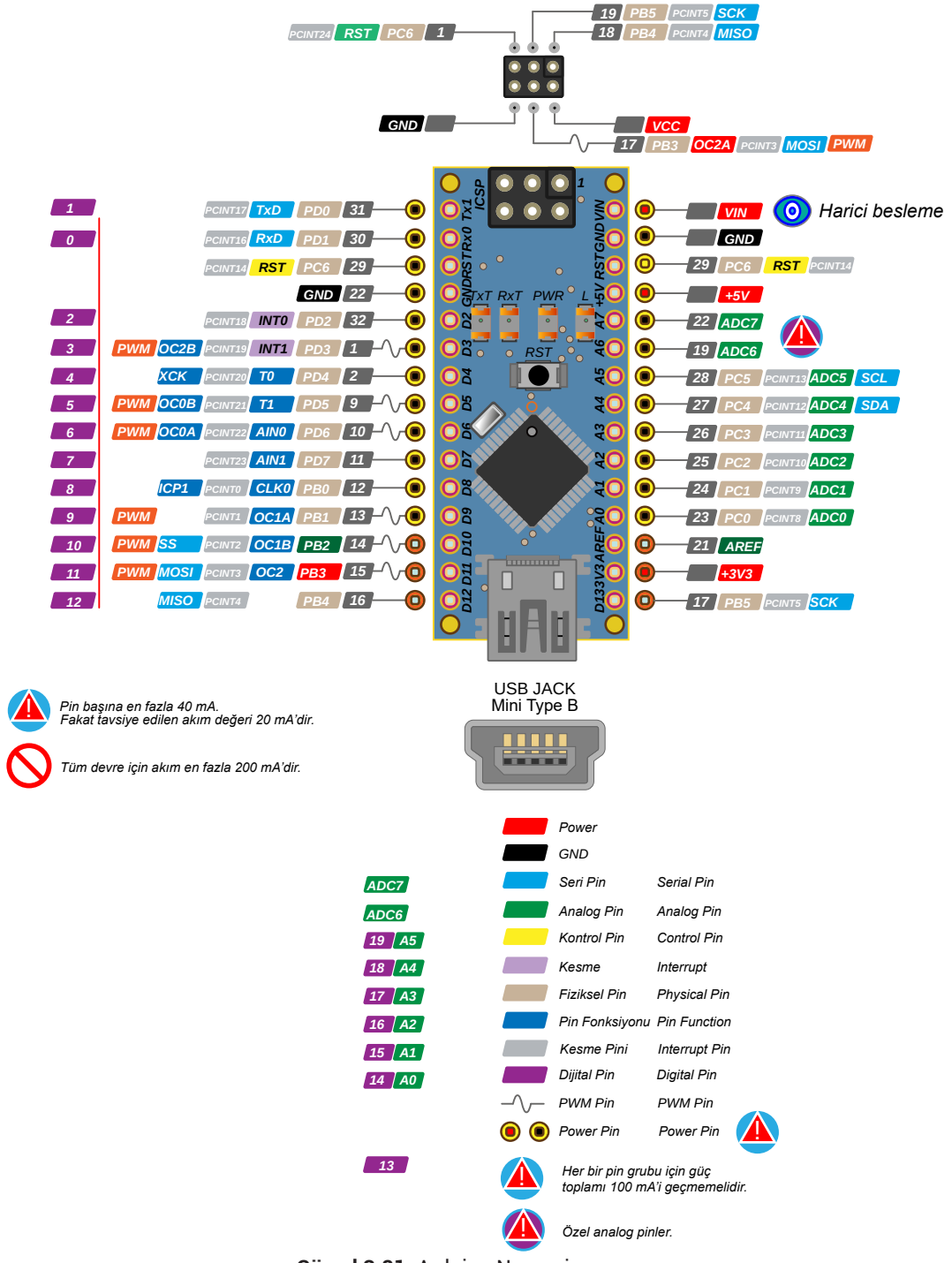
Arduino Uno ve Nano'da SPI haberleşme pinleri, 11-MOSI, 12-MISO ve 13-SCK pinleridir (Görsel 3.80).



Görsel 3.80: Arduino Uno ve Nano'da SPI haberleşme pin numaraları

Arduino Nano pin yapısı

Verici uygulamalarında küçük boyutundan dolayı Arduino Nano tercih edilmiştir. Görsel 3.81'de Arduino Nano pin yapısı verilmiştir.

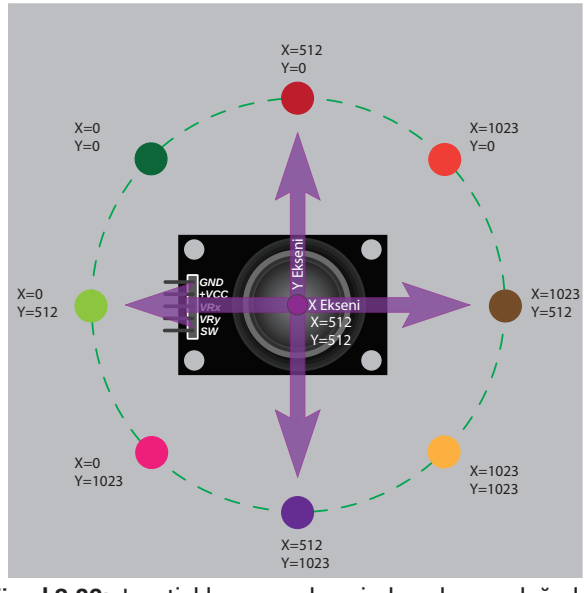


Görsel 3.81: Arduino Nano pin yapısı

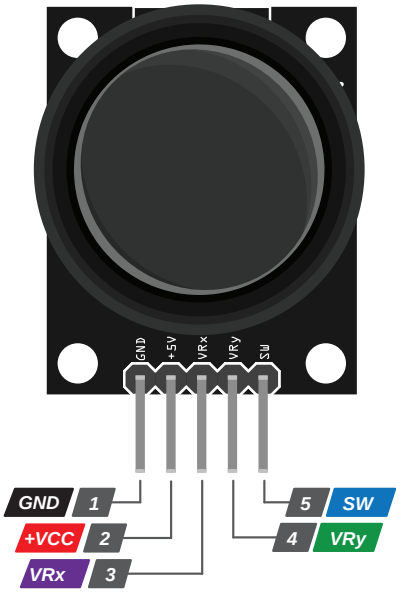
Uygulama Adı: Uzaktan Kontrollü Araba Uygulaması No.: 17

Amaç: Uzaktan kontrollü araba uygulaması yapmak.

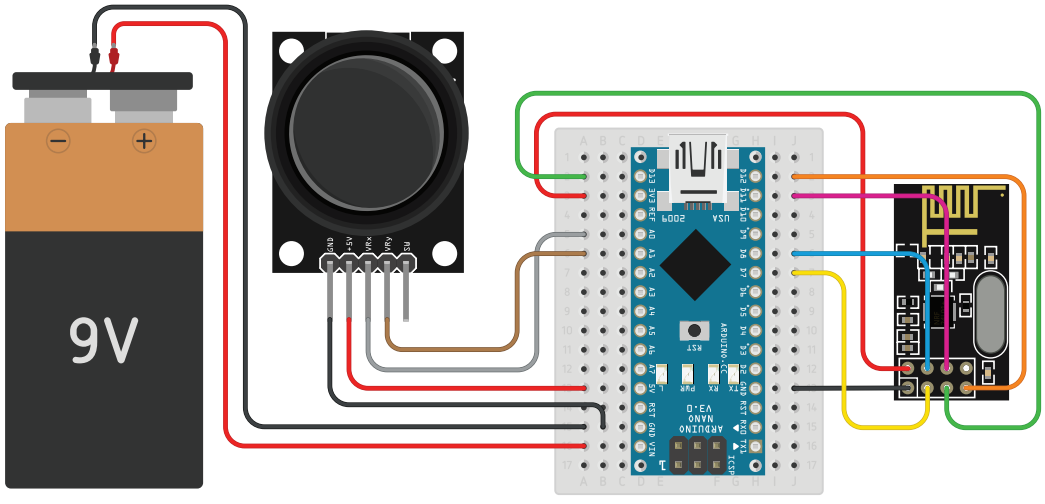
Görsel 3.82'de **joystick** modülünün 10 bitlik ADC'de aldığı değerler verilmiştir. Görsel 3.83'te **joystick** modülünün pin yapısı görülmektedir. Görsel 3.84'teki verici ve Görsel 3.85'teki alıcı devrelerinde birer adet nRF24L01+ modül kullanılarak kablosuz iletişim uygulaması yapılmıştır. nRF24L01+ modülü Arduino ile haberleşirken SPI protokolünü kullanır. **Joystick**le x ve y ekseninde 0-1023 arasında okunan değerler map() foksiyonuyla -255 ile 255 arası değerlere dönüştürülerek alıcıya gönderilir. Alıcıda negatif değerler "-" ile çarpılıp pozitif değer PWM olarak geri ve sol dönüşe uygulanır.



Görsel 3.82: Joystickle x ve y ekseninden okunan değerler



Görsel 3.83: Joystick modülü pin yapısı



Görsel 3.84: Verici uygulaması

Uzaktan kontrollü araba verici uygulaması programı aşağıdaki gibidir:

```
#include <SPI.h> // nRF24L01+ ile SPI iletişim.
#include <RF24.h> // nRF24L01+ için RF24 kütüphanesi v1.4.1

int x, y; // x ve y düzlemlerinin değişkenleri.

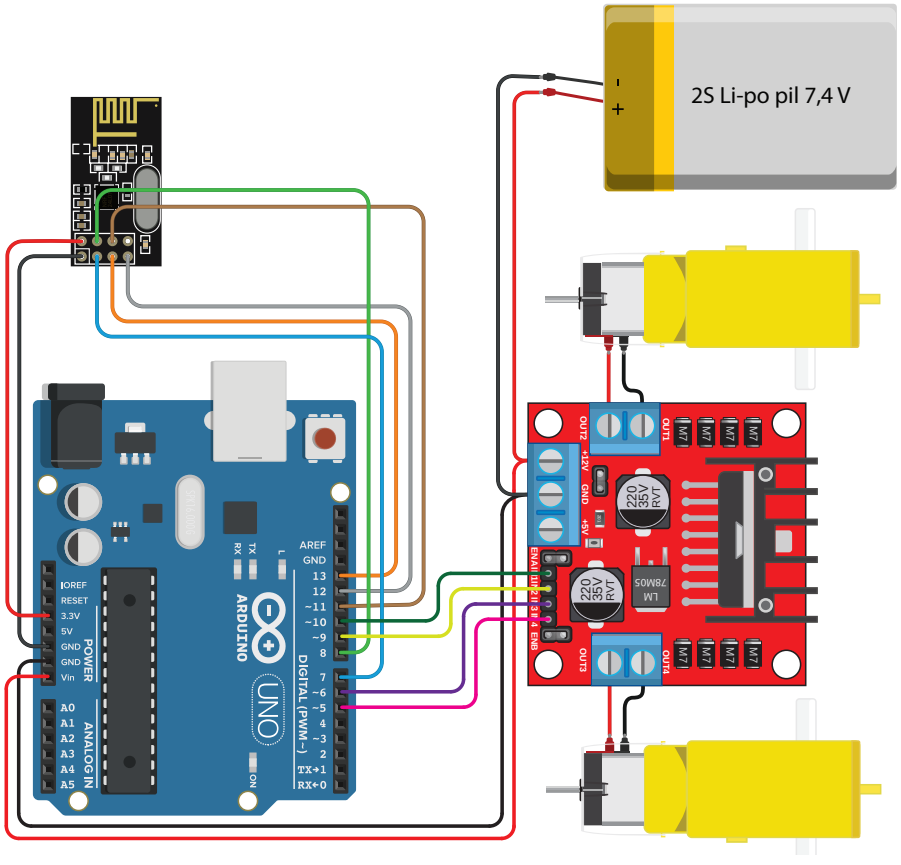
RF24 radio(7, 8); // radio isimli nesne oluştur. CE ve CSN pinleri.
int veri[2]; // X ve y düzlemi için dizi tanımlama.

void setup() {
  radio.begin(); // nRF24L01+'yı başlat.
  radio.openWritingPipe(1234); // İletişim anahtarı 1234.
}

void loop() {
  x = analogRead(A0); // joystick x ekseni bilgisini al.
  y = analogRead(A1); // joystick y ekseni bilgisini al.

  veri[0] = map(x, 0, 1023, -255, 255 ); //X düzleminin verisi (Geri-ileri).
  veri[1] = map(y, 0, 1023, -255, 255); //Y düzeleminin verisi (Sol-sağ).

  radio.write(veri, sizeof(veri)); // Veri değişkenindeki bilgiyi nRF24L01+'ya gönder.
}
```



Görsel 3.85: Alıcı uygulaması

Uzaktan kontrollü araba alıcı uygulaması programı aşağıdaki gibidir:

```
#include <SPI.h> // nRF24L01+ ile SPI iletişim.
#include <RF24.h> //  v1.4.2

RF24 radio(7, 8); // radio isimli nesne oluştur. CE ve CSN pinleri.

int veri[2];// X ve Y düzlemi için dizi tanımlama

const byte IN1 = 10, IN2 = 9, IN3 = 6, IN4 = 5; // PWM pinleri. (enA ve enB'de jumper takılı).
// IN1 ve IN2 sol tekerlek, IN3 ve IN4 sağ tekerlek.
void setup() {

  Serial.begin(9600); // Hataları seri monitörde görebilmek için seri iletişimi başlat.
  if (!radio.begin()) { // nRF24L01+'yı başlat.

    Serial.println("nRF24L01+ bağlantıları yanlış veya nRF24L01+ bozulmuş olabilir.");

//nRF24L01+ modülü başlamıyorsa uyarı ver.
    while (1) {}

  }
```

```
 radio.openReadingPipe(0, 1234); // İletişim anahtarı 1234.
  radio.startListening(); // Alıcı moda geç.
}

void loop() {
  if (radio.available()) { // Veri geliyorsa...
    radio.read(&veri, sizeof(veri)); // Gelen veriyi veri değişkenine yükle.

    if (veri[0] > 50) { //İleri.
      analogWrite(IN1, veri[0]);
      analogWrite(IN2, 0);
      analogWrite(IN3, veri[0]);
      analogWrite(IN4, 0);
    }
    if (veri[0] < -50) { //Geri.
      analogWrite(IN1, 0);
      analogWrite(IN2, -veri[0]);
      analogWrite(IN3, 0);
      analogWrite(IN4, -veri[0]);
    }
    if (veri[1] > 50) { // Sağ.
      analogWrite(IN1, veri[1]);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, veri[1]);
    }
    if (veri[1] < -50) { //Sol.
      analogWrite(IN1, 0);
      analogWrite(IN2, -veri[1]);
      analogWrite(IN3, -veri[1]);
      analogWrite(IN4, 0);
    }
    if (veri[0] > -50 && veri[0] < 50 && veri[1] > -50 && veri[1] < 50) { // Dur.
      analogWrite(IN1, 0);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, 0);
    }
  }
}
```

İşlem Basamakları

1. Görsel 3.84'teki verici ve Görsel 3.85'teki alıcı devreyi kurunuz. Programları yükleyiniz.
2. Verici devresindeki **joystick**i kullanarak alıcı aracın çalışmasını gözlemleyiniz.

SIRA SİZDE

1. **Joystick** üzerine basınca (sw) araçta korna (**buzzer**) çalan düzenlemeyi gerçekleştiriniz.

Verici kodu

```
#include <SPI.h> // nRF24L01+ ile SPI iletişim.
#include <RF24.h> //  v1.4.2

int x, y; // x ve y düzlemlerinin değişkenleri.

RF24 radio(7, 8); // radio isimli nesne oluştur. CE ve CSN pinleri.
int veri[3]; // x, y düzlemi ve korna (sw) için dizi tanımlama.
const byte sw=2; // sw (buton) pini.

void setup() {
  pinMode(sw, INPUT_PULLUP); // Butona basınca 0.

  Serial.begin(9600); // Hataları seri monitörde görebilmek için seri iletişimi başlat.
  if (!radio.begin()) { // nRF24L01+'yı başlat.

    Serial.println("nRF24L01+ bağlantıları yanlış veya temazsızlık var. nRF24L01+ bozulmuş olabilir."); //nRF24L01+ modülü başlamıyorsa uyarı ver.
    while (1) {}
  }
  radio.openWritingPipe(1234); // İletişim anahtarı 1234.
}

void loop() {
  x = analogRead(A0);
  y = analogRead(A1);
  bool swDurum = digitalRead(sw); // Butonun durumunu oku.

  veri[0] = map(x, 0, 1023, -255, 255 ); //X düzleminin verisi (Geri-ileri).
  veri[1] = map(y, 0, 1023, -255, 255); //Y düzeleminin verisi (Sol-sağ).
  veri[2] = swDurum; //Butona basılma bilgisi. 0 basıldı, 1 basılmadı bilgisi.


  radio.write(veri, sizeof(veri)); // veri değişkenindeki bilgiyi nRF24L01+'ya gönder.
}
```

Alıcı kodu

```
#include <SPI.h> // nRF24L01+ ile SPI iletişim.
#include <RF24.h> //  v1.4.2

RF24 radio(7, 8); // radio isimli nesne oluştur. CE ve CSN pinleri.
```

```
int veri[3];// X, y düzlemi ve sw için dizi tanımlama.

const byte IN1 = 5, IN2 = 6, IN3 = 9, IN4 = 10; // PWM pinleri. (enA ve enB'de jumper takılı).
// IN1 ve IN2 sol tekerlek, IN3 ve IN4 sağ tekerlek.
const byte buzzer = 11; // Buzzer pin.

void setup() {

  Serial.begin(9600); // Hataları seri monitörde görebilmek için seri iletişimi başlat.
  if (!radio.begin()) { // nRF24L01+'yı başlat.

    Serial.println("nRF24L01+ bağlantıları yanlış veya nRF24L01+ bozulmuş olabilir."); //
nRF24L01+ modülü başlamıyorsa uyarı ver.
    while (1) {}
  }
  radio.openReadingPipe(0, 1234); // İletişim anahtarı 1234.
  radio.startListening(); // Alıcı moda geç.
  pinMode(buzzer, OUTPUT);
}

void loop() {
  if (radio.available()) { // Veri geliyorsa...
    radio.read(&veri, sizeof(veri)); // Gelen veriyi veri değişkenine yükle.

    if (veri[0] > 50) { //İleri.
      analogWrite(IN1, veri[0]);
      analogWrite(IN2, 0);
      analogWrite(IN3, veri[0]);
      analogWrite(IN4, 0);
    }
    if (veri[0] < -50) { //Geri.
      analogWrite(IN1, 0);
      analogWrite(IN2, -veri[0]);
      analogWrite(IN3, 0);
      analogWrite(IN4, -veri[0]);
    }
    if (veri[1] > 50) { // Sağ.
      analogWrite(IN1, veri[1]);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, veri[1]);
 }
    if (veri[1] < -50) { //Sol.
      analogWrite(IN1, 0);
      analogWrite(IN2, -veri[1]);
```

```
      analogWrite(IN3, -veri[1]);
      analogWrite(IN4, 0);
    }
    if (veri[0] > -50 && veri[0] < 50 && veri[1] > -50 && veri[1] < 50) { // Dur.
      analogWrite(IN1, 0);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, 0);
    }
    if (veri[2] == 0)
      digitalWrite(buzzer, HIGH); // Korna çal.
    else
      digitalWrite(buzzer, LOW); // Korna durdur.
  }
}
```

2. Aracı ileri-sağ, ileri-sol, geri-sağ ve geri-sol yönlere çapraz hareket edebilecek şekilde programı düzenleyiniz.

```
 #include <SPI.h> // nRF24L01+ ile SPI iletişim.
#include <RF24.h> //  v1.4.2

RF24 radio(7, 8); // radio isimli nesne oluştur. CE ve CSN pinleri.

byte solPwm, sagPwm;
int veri[2];// X ve Y düzlemi için dizi tanımlama

const byte IN1 = 5, IN2 = 6, IN3 = 9, IN4 = 10; // PWM pinleri. (enA ve enB'de jumper takılı).
// IN1, IN2 sol tekerlek, IN3, IN4 sağ tekerlek.

void setup() {

  Serial.begin(9600); // Hataları seri monitörde görebilmek için seri iletişimi başlat.
  if (!radio.begin()) { // nRF24L01+'yı başlat.

    Serial.println("nRF24L01+ bağlantıları yanlış veya nRF24L01+ bozulmuş olabilir."); //
nRF24L01+ modülü başlamıyorsa uyarı ver.
    while (1) {}
  }
  radio.openReadingPipe(0, 1234); // İletişim anahtarı 1234.
  radio.startListening(); // Alıcı moda geç.
}

void loop() {
  if (radio.available()) { // Veri geliyorsa...

    radio.read(&veri, sizeof(veri)); // Gelen veriyi veri değişkenine yükle.
    int x = veri[0]; // x'in içeriği -255 ile 255 arasında değişmektedir.
```

```
    int y = veri[1]; // y'in içeriği -255 ile 255 arasında değişmektedir.

    if (y > 0) { // y pozitif ise araç sağa döner.
      solPwm = x; // Sol tekerlek hızını x ekseninden gelen sayı yap.
      sagPwm = x - y / 5; // Sağ tekerlek hızını (pwm) y ekseninden gelen sayının 1/5'i kadar
azalt.
    }
    else { // y negatif ise araç sola döner.
      sagPwm = x;
      solPwm = x + y / 5; // Solda y negatiftir (Sol tekerleğin hızını azalt).
    }

    if (x > 50) { // İleri yönler.
      analogWrite(IN1, solPwm);
      analogWrite(IN2, 0);
      analogWrite(IN3, sagPwm);
      analogWrite(IN4, 0);
    }
    if (x < -50) { // Geri yönler.
      analogWrite(IN1, 0);
      analogWrite(IN2, 255 - solPwm); // Joystick'i geri çektikçe geriye doğru hızlan.
      analogWrite(IN3, 0);
      analogWrite(IN4, 255 - sagPwm);
    }
    if (x > -50 && x < 50 && y > -50 && y < 50) { // Joystick sabitse dur.
      analogWrite(IN1, 0);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, 0);
    }
  }
}
```

Sorular

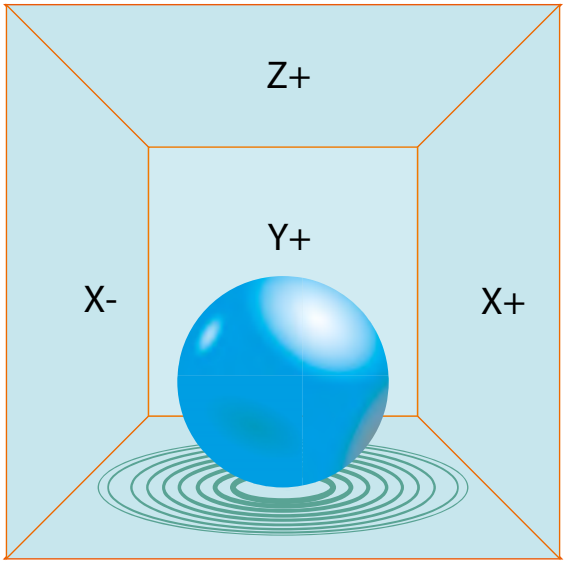
1. **Joystick** modülü ile potansiyometre arasındaki fark nedir? Açıklayınız.
2. veri[0] hangi eksenin (x, y) bilgisini içermektedir? Belirtiniz.
3. **Joystick** sabitken x ve y'nin değeri nedir? Belirtiniz.
4. "if (veri[0] > 50) { //İleri." satırında neden 0 yerine 50 kullanılmıştır? Açıklayınız.
5. veri[0] değişkeni negatif bir sayı içeriyorsa -veri[0] ile abs(veri[0]) aynı sonucu verir mi? Nedeni ile açıklayınız.

İvme ve Eğim Sensörü

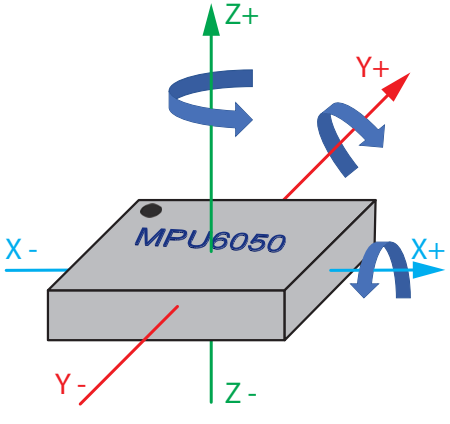
MPU6050, üzerinde üç eksenli gyro (Görsel 3.86) ve üç eksenli açısal ivmeölçer (Görsel 3.87) bulunduran IMU [Inertial Measurement Unit (inırşıl mejırmınt yunıt) Atalet ölçü birimi] sensör kartıdır. IMU sensörleri, denge robotları, uçaklar, cep telefonları, tabletler, uzay araçları, uydular, hava aracı vb. araçlarda konum tespiti, yön bulma, hareket izleme ve uçuş kontrolü için kullanılır. İvmeölçer doğrusal ivmeyi ölçerken jiroskop açısal dönüşü (°/s) ölçer.

MPU6050 sensör kartı üzerinde voltaj regulatörü bulunduğundan 3 V ile 5 V arası bir besleme voltajı ile çalıştırılabilir (Görsel 3.88). İvmeölçer ve gyro çıkışlarının her ikisi de ayrı kanallardan I²C çıkışı vermektedir. Her eksende 16 bitlik bir çözünürlükle çıkış verebilmektedir.

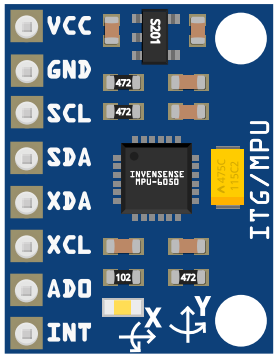
İvmeölçer üzerine düşen statik (yer çekimi) veya dinamik (aniden hızlanma veya durma) ivmeyi ölçmektedir. Görsel 3.86'da çalışma prensibi verilen sensöre 1g'lık bir yer çekimi kuvveti etki etmektedir. Bu kuvvet de yaklaşık olarak 9.8 m/sn.dir. Bu durumda MPU6050 kütüphanesi Z değerini 16384 olarak verir. Sensör ters çevrildiğinde Z değeri -16384 olur. X ve y eksenleri için de aynı durum geçerlidir. Kartın açılı olması durumunda bulunduğu konuma göre X, Y ve Z değerleri -16384 ile 16384 arasında değer alır.



Görsel 3.86: İvmeölçer çalışma prensibi



Görsel 3.87: Hassasiyet eksenlerinin yönü ve dönme polaritesi



Görsel 3.88: MPU6050 pin yapısı

VCC: 5 V – 35 V besleme.

GND: Topraklama pini.

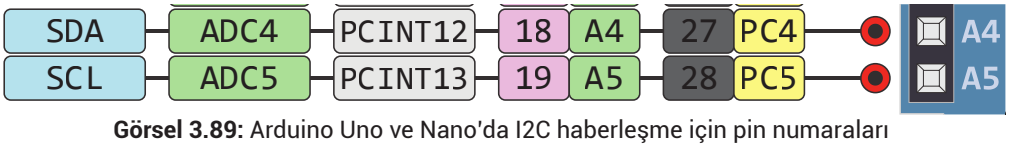
SCL: I2C saat pinidir. **Master** cihaz tarafından sağlanan bir zamanlama sinyalidir.

SDA: I2C veri pinidir. Bu hat hem gönderme hem de alma için kullanılır.

XDA: Haricî I2C veri hattıdır. Haricî I2C veri yolu, haricî sensörleri bağlamak içindir.

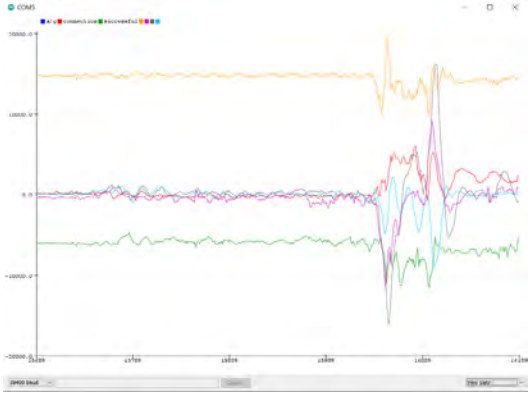
AD0: MPU6050 modülünün dâhilî I2C adresini değiştirmenizi sağlar. Modül başka bir I2C cihazıyla çakışıyorsa veya aynı I2C veri yolunda iki MPU6050 kullanılırsa bu pin kullanılır. AD0 pini boş bırakıldığında varsayılan I2C adresi 0x68 **HEX** olur. 3,3 V'a bağlandığında I2C adresi 0x69 **HEX** olur.

Arduino Uno ve Nano'da I2C haberleşme pinleri, A4 ve A5 analog giriş pinleridir. A4-SDA ve A5-SCL'dir (Görsel 3.89).



Görsel 3.89: Arduino Uno ve Nano'da I2C haberleşme için pin numaraları

Sensörü test etmek için bağlantılar yapıldıktan ve MPU6050 kütüphanesi yüklendikten sonra Arduino IDE menülerinden "Dosya →Örnekler→MPU6050→ MPU6050_raw" örneği açılır. Altı eksenin değeri seri monitörde gözlenir. Değerleri daha rahat izleyebilmek için araçlar menüsünden seri çizici seçilerek Serial.print() fonksiyonuyla yazdırılan değerler grafik olarak gözlemlenebilir (Görsel 3.90).



Görsel 3.90: Değerleri seri çizicide gösterme

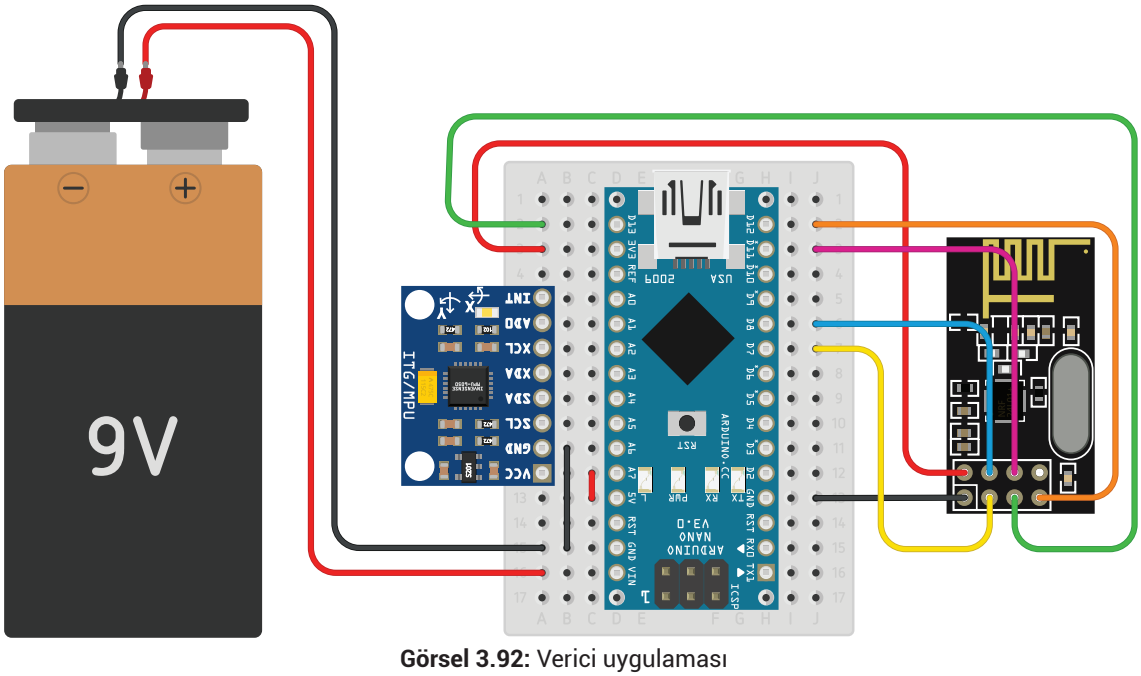
Uygulama Adı: Elle Uzaktan Kontrollü Araba Uygulaması No.: 18

Amaç: Elle uzaktan kontrollü araba uygulaması yapmak.

Görsel 3.91'de eldiven üzerine yerleştirilmiş verici devresi görülmektedir. Görsel 3.92'deki vericide kullanılan MPU6050 sensörü küçük deney borduna SDA pini A4, SCL pini A5'e gelecek şekilde takılmıştır. Görsel 3.93'teki alıcı devresinde nRF24L01+ modül kullanılarak kablosuz iletişim uygulaması yapılmıştır. nRF24L01+ modülü Arduino ile haberleşirken SPI protokolünü, MPU6050 ivme sensörü I2C protokolünü kullanmaktadır. İvme sensörünün x ve y ekseninde -16384 ile 16384 arasında ürettiği değerler map() foksiyonuyla -255 ile 255 arası değerlere dönüştürülerek alıcıya gönderilir. Alıcıda negatif değerler "-" ile çarpılarak pozitif değer PWM olarak geri ve sağ dönüşe uygulanır.



Görsel 3.91: Vericiyi eldivenle kontrol etme



Görsel 3.92: Verici uygulaması

Elle uzaktan kontrollü araba verici uygulaması programı aşağıdaki gibidir:

```
#include <SPI.h> // nRF24L01+ ile SPI iletişim.
#include <RF24.h> // nRF24L01+ kütüphanesi v1.4.1
#include <Wire.h> // Mpu6050 ile I2C iletişim.
#include <MPU6050.h> // Mpu6050 kütüphanesi v0.3.0

MPU6050 ivme_sensor; // ivme_sensor isimli nesne oluştur.
int x, y, z; // x, y, z düzlemlerinin ivme değişkenleri.

RF24 radio(7, 8); // radio isimli nesne oluştur. CE ve CSN pinleri.
int veri[2]; // X ve y düzlemi için dizi tanımlama.

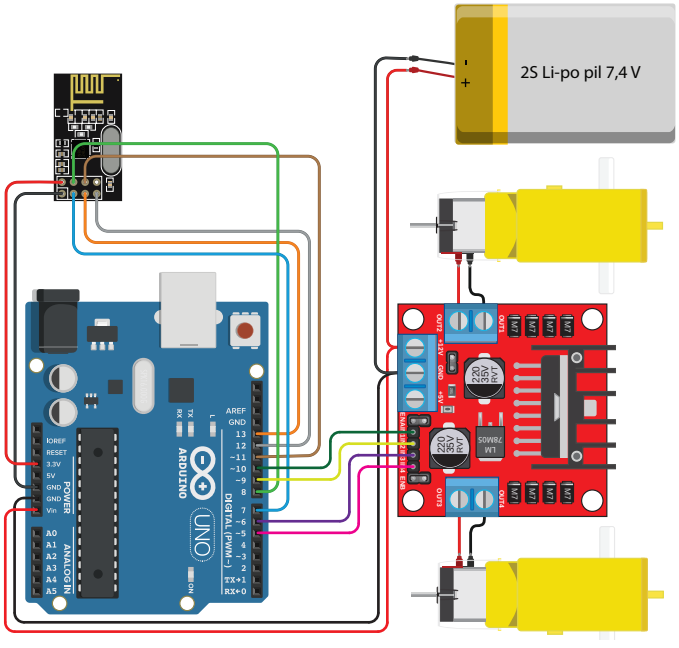
void setup() {
  Wire.begin(); // I2C iletişimi başlat.
  ivme_sensor.initialize(); // Mpu6050'yi başlat.

  radio.begin(); // nRF24L01+'yı başlat.
  radio.openWritingPipe(1234); // İletişim anahtarı 1234.
}

void loop() {
  ivme_sensor.getAcceleration(&x, &y, &z); // Sensörden ivme bilgilerini al.

  veri[0] = map(x, -16384, 16384, -255, 255 ); // X düzleminin verisi (ileri-geri)
  veri[1] = map(y, -16384, 16384, -255, 255); // Y düzeleminin verisi (sağ-sol)

  radio.write(veri, sizeof(veri)); // Veri değişkenindeki bilgiyi nRF24L01+'ya gönder.
}
```



Görsel 3.93: Alıcı uygulaması

Elle uzaktan kontrollü araba alıcı uygulaması programı aşağıdaki gibidir:

```
#include <SPI.h>
#include <RF24.h> // v1.4.1

RF24 radio(7, 8); // radio isimli nesne oluştur. CE ve CSN pinleri.

int veri[2];// X ve Y düzlemi için dizi tanımlama
const byte IN1 = 5, IN2 = 6, IN3 = 9, IN4 = 10; // PWM pinleri. (enA ve enB'de jumper takılı).

void setup() {
  radio.begin(); // NRF'yi başlat
  radio.openReadingPipe(0, 1234); // İletişim anahtarı 1234.
  radio.startListening(); // Alıcı moda geç.
}

void loop() {
  if (radio.available()) { // Veri geliyorsa...
    radio.read(veri, sizeof(veri)); // Gelen veriyi veri değişkenine yükle.

    if (veri[0] > 50) { //İleri.
      analogWrite(IN1, veri[0]);
      analogWrite(IN2, 0);
      analogWrite(IN3, veri[0]);
      analogWrite(IN4, 0);
    }
    if (veri[0] < -50) { //Geri.
      analogWrite(IN1, 0);
      analogWrite(IN2, -veri[0]);
      analogWrite(IN3, 0);
      analogWrite(IN4, -veri[0]);
    }
    if (veri[1] > 50) { //Sol.
      analogWrite(IN1, 0);
      analogWrite(IN2, veri[1]);
      analogWrite(IN3, veri[1]);
      analogWrite(IN4, 0);
    }
    if (veri[1] < -50) { // Sağ.
      analogWrite(IN1, -veri[1]);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, -veri[1]);
    }
```

```
  if (veri[0] > -50 && veri[0] < 50 && veri[1] > -50 && veri[1] < 50) { // Dur.
   analogWrite(IN1, 0);
   analogWrite(IN2, 0);
   analogWrite(IN3, 0);
   analogWrite(IN4, 0);
  }
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.92'deki verici ve Görsel 3.93'teki alıcı devreyi kurunuz. Programları yükleyiniz.
4. Verici devresine x ve y eksenlerinde eğim vererek alıcı aracın çalışmasını gözlemleyiniz.
5. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Sensör titreştirildiğinde alıcıda korna (**buzzer**) çalan uygulamayı gerçekleştiriniz.

Verici kodu

```
#include <SPI.h> // nRF24L01+ ile SPI iletişim.
#include <RF24.h> // v1.4.2
#include <Wire.h> // Mpu6050 ile I2C iletişim.
#include <MPU6050.h> //Mpu6050 kütüphanesi v0.3.0

MPU6050 ivme_sensor; // ivme_sensor isimli nesne oluştur.
int ax, ay, az, gx, gy, gz; // x, y, z düzlemlerinin ivme değişkenleri.

RF24 radio(7, 8); // radio isimli nesne oluştur. CE ve CSN pinleri.
int veri[3]; // X ve y düzlemi için dizi tanımlama.

void setup() {
  Wire.begin(); // I2C iletişimi başlat.
  ivme_sensor.initialize(); // Mpu6050'yi başlat.


  Serial.begin(9600); // Hataları seri monitörde görebilmek için seri iletişimi başlat.
  if (!radio.begin()) { // nRF24L01+'yı başlat.
    Serial.println("nRF24L01+ bağlantıları yanlış veya temazsızlık var. nRF24L01+ bozulmuş olabilir."); //nRF24L01+ modülü başlamıyorsa uyarı ver.
    while (1) {}
  }
  radio.openWritingPipe(1234); // İletişim anahtarı 1234.
```

```
}

void loop() {
  ivme_sensor.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);  // Sensörden ivme bilgilerini al.

  veri[0] = map(ax, -16384, 16384, -255, 255 ); //X düzleminin verisi (ileri-geri)
  veri[1] = map(ay, -16384, 16384, -255, 255); //Y düzleminin verisi (sağ-sol)
  veri[2] = gx;
  radio.write(&veri, sizeof(veri)); // Veri değişkenindeki bilgiyi nRF24L01+'ya gönder.
}
```

Alıcı kodu

```
#include <SPI.h> // nRF24L01+ ile SPI iletişim.
#include <RF24.h> //  v1.4.2

RF24 radio(7, 8); // radio isimli nesne oluştur. CE ve CSN pinleri.

int veri[3];// X ve Y düzlemi için dizi tanımlama
const byte IN1 = 5, IN2 = 6, IN3 = 9, IN4 = 10; // PWM pinleri. (enA ve enB'de jumper takılı).
const byte buzzer = 4;
void setup() {
  Serial.begin(9600); // Hataları seri monitörde görebilmek için seri iletişimi başlat.
  if (!radio.begin()) { // nRF24L01+'yı başlat.
    Serial.println("nRF24L01+ bağlantıları yanlış veya nRF24L01+ bozulmuş olabilir.");
 //nRF24L01+ modülü başlamıyorsa uyarı ver.
    while (1) {}
  }
  radio.openReadingPipe(0, 1234); // İletişim anahtarı 1234.
  radio.startListening(); // Alıcı moda geç.
  pinMode(buzzer, OUTPUT); // Buzzer pin.
}

void loop() {
  if (radio.available()) { // Veri geliyorsa...
    radio.read(&veri, sizeof(veri)); // Gelen veriyi veri değişkenine yükle.

    if (veri[0] > 50) { //İleri.
      analogWrite(IN1, veri[0]);
      analogWrite(IN2, 0);
      analogWrite(IN3, veri[0]);
      analogWrite(IN4, 0);
    }
    if (veri[0] < -50) { //Geri.
```

```
      analogWrite(IN1, 0);
      analogWrite(IN2, -veri[0]);
      analogWrite(IN3, 0);
      analogWrite(IN4, -veri[0]);
    }
    if (veri[1] > 50) { //Sol.
      analogWrite(IN1, 0);
      analogWrite(IN2, veri[1]);
      analogWrite(IN3, veri[1]);
      analogWrite(IN4, 0);
    }
    if (veri[1] < -50) { // Sağ.
      analogWrite(IN1, -veri[1]);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, -veri[1]);
    }
    if (veri[0] > -50 && veri[0] < 50 && veri[1] > -50 && veri[1] < 50) { // Dur.
      analogWrite(IN1, 0);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, 0);
    }
    if (veri[2] > 16000) {
      digitalWrite(buzzer, HIGH); // Korna çal.
      delay(200);
    }
    else
      digitalWrite(buzzer, LOW); // Korna durdur.
  }
}
```

Soru

1. MPU6050 ile denge robotu nasıl yapılır? Araştırınız.

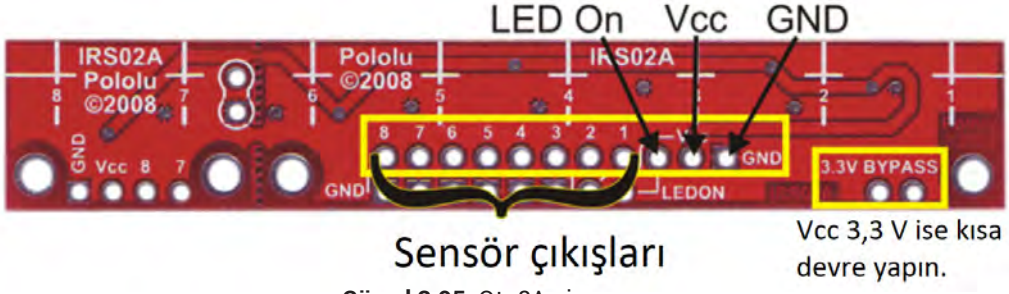
Çizgi İzleyen Sensör

Çizgi izleyen robotlarda kullanılan sensörler kızılötesi verici ve kızılötesi alıcı çiftinden oluşur. Vericiden çıkan kızılötesi ışık beyaz yüzeye çarptığında yansır. Alıcı, yansıyan bu ışığı algılar. Siyah zemin kızılötesi ışığı yansıtmayacağı için alıcıda algılama gerçekleşmez. Görsel 3.94'teki QTR8A sensöründe kızılötesi sensör çiftinden sekiz adet bulunmaktadır. Analog çıkış veren bu sensör grubunda beyaz zemine denk gelen sensör 0 V'a yakın değerler verirken siyah zemine denk gelen sensörler 5 V'a (Vcc) yakın değerler verir.



Görsel 3.94: Qtr 8A sensör dizilimi

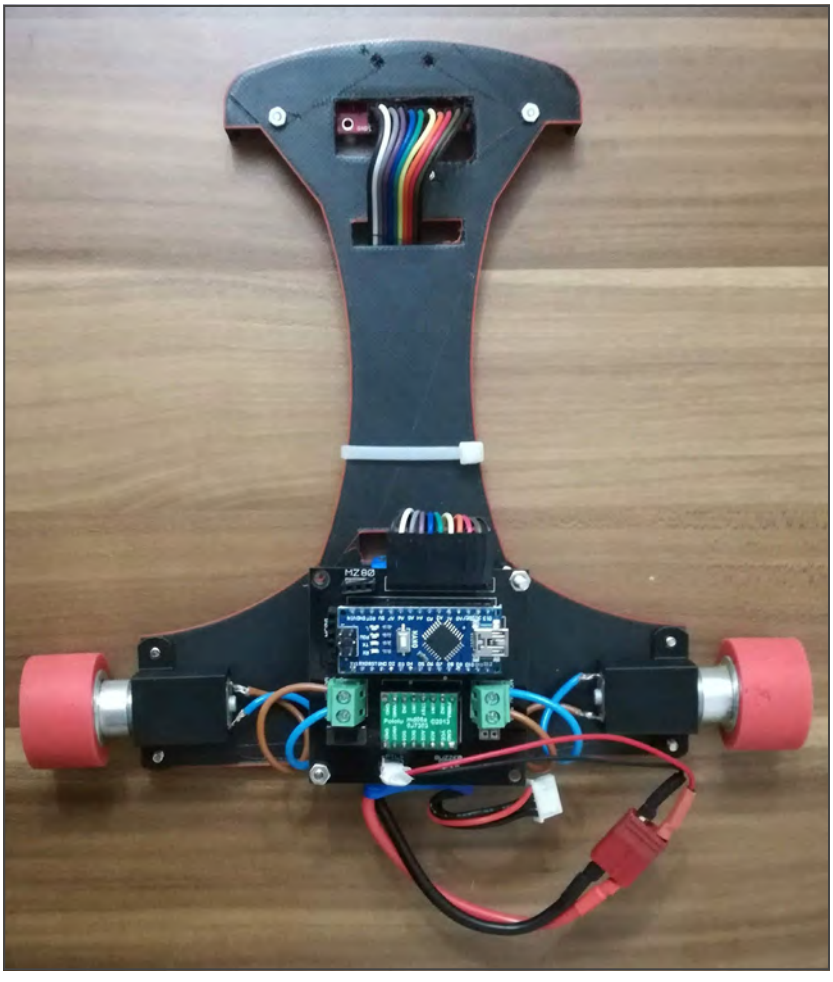
Görsel 3.95'teki sensör kartı 3,3 V-5 V gerilimle çalışmaktadır ve 100 mA akım çekmektedir. En iyi algılama mesafesi 3 mm'dir. Sensörün zeminden uzaklığı 6 mm'yi geçmemelidir. "LED On" pini lojik 1 olduğunda veya boş bırakıldığında (Dâhilî pull-up bağlıdır.) verici LED'ler çalışmaktadır. Lojik 0 olduğunda verici LED'ler kapalıdır.



Görsel 3.95: Qtr 8A pin yapısı

Bu kartla birlikte kullanılan QTRSensors.h kütüphanesi farklı sayıda kızılötesi sensör içeren diğer sensör kartlarında da kullanılabilir.

Çizgi izleyen robotlar çizgiyi takip edecek şekilde programlanır. QTRSensors.h kütüphanesi sensör sayısına göre her bir sensörden gelen değerlerin ağırlıklı ortalamasını alarak çizginin bulunduğu pozisyonu verir. Çizginin pozisyonu hedef değerden farklıysa kapalı çevrim kontrol sistemi komutlarıyla (**pid**) pozisyon değeri hedef değere yaklaşacak şekilde motor pwm değerleri değiştirilerek araç çizgide yol alır. Çizgi izleyen robotta dışarıdan bir müdahale olmadığından (rüzgâr, dalga, arazi yolu) **pid** sisteminde integral kullanılmasına gerek yoktur. **Pid** sisteminde kullanılan katsayılar aracın boyuna ve ağırlığına göre değişmektedir (Görsel 3.96).

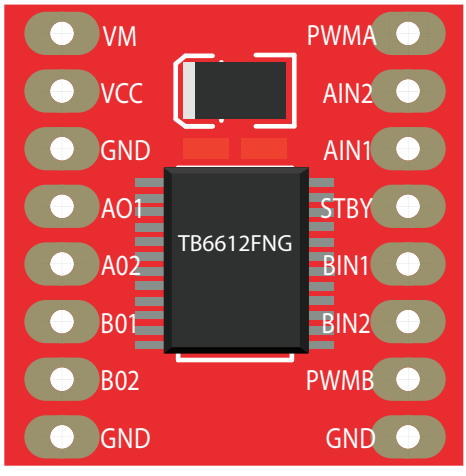


Görsel 3.96: Çizgi izleyen robot

Uygulama Adı: Çizgi İzleyen Robot Uygulaması No.: 19

Amaç: Çizgi izleyen robot uygulaması yapmak.

Görsel 3.97'de Sparkfun TB6612fng motor sürücü kartı pin yapısı verilmiştir. Görsel 3.98'deki çizgi izleyen robot devresinde Sparkfun TB6612fng (kırmızı renkli PCB) motor sürücü kartı kullanılmıştır. Pololu TB6612fng entegresini kullanan yeşil renkli PCB kartın boyutları ve pin yerleşim düzeni farklıdır. Çizgi izleyen robot programı L298N motor sürücüsüyle de kullanılabilir.



Görsel 3.97: TB6612fng pin yapısı

VCC: 5 V besleme.

GND: Topraklama pini.

VM: Motor besleme girişi 2,5 V-13,5 V.

STBY: "LOW"=standby. Sürücüyü aktif yapmak için lojik 1 yapılır. 5 V hattına bağlanılarak da kullanılabilir.

AO1, AO2: Birinci motor (sağ) çıkışları. 1,2 A sürekli, 3,2 A anlık.

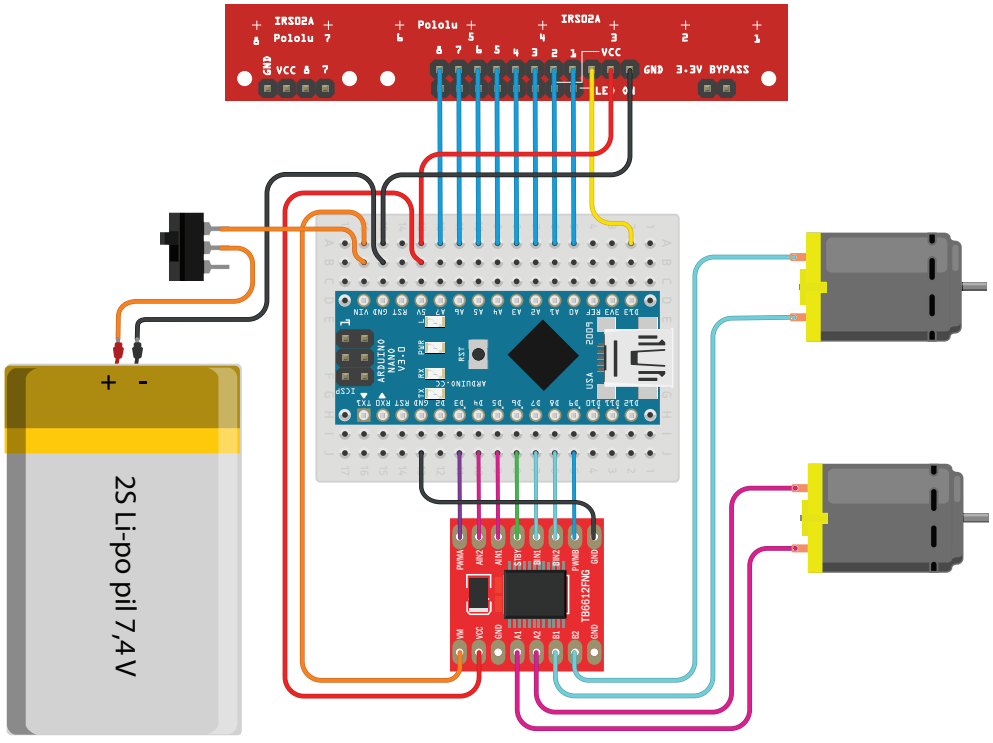
BO1, BO2: İkinci motor (sol) çıkışları. 1,2 A sürekli, 3,2 A anlık.

AIN1, AIN2: Birinci motor (sağ) kontrol girişleri. 200 kΩ dâhilî pull-down.

BIN1, BIN2: İkinci motor (sol) kontrol girişleri. 200 kΩ dâhilî pull-down.

PWMA: Birinci motor (sağ) PWM girişi.

PWMB: İkinci motor (sol) PWM girişi.



Görsel 3.98: Çizgi izleyen robot uygulaması

Çizgi izleyen robot uygulaması programı aşağıdaki gibidir:

```
#include <QTRSensors.h> //Qtr v4.0

#define PWMA 3 // A sağ motor.
#define AIN2 4
#define AIN1 5
#define STBY 6
#define BIN1 7 // B sol motor.
#define BIN2 8
#define PWMB 9

#define sensorSayisi 8
#define sensorOrnekSayisi 4
#define emiterPini 11
#define LED 13

int maxHiz = 70; // Motor pwm ayarı 0 - 255.

int hata = 0, turev = 0;

float KP = 0.03, KD = 0.5; // Oran (KP) ve türev (KD) sabitleri. (Her araca göre ayar yapılmalıdır.)
unsigned int pozisyon = 3500;
```

```
int fark = 0; // Motorlara uygulanan fark.
int sonHata; // Orantılı son değer. (Hatanın türevini hesaplamak için kullanılır.)
int hedef = 3500; // Sensörden gelen 0 - 7000 arası değerin orta noktası.

QTRSensors qtr; // qtr isimli nesne oluşturuldu.
unsigned int sensor[sensorSayisi];

void setup() {

  qtr.setTypeAnalog();  //QTR-8A ayarla. (QTR-8RC için qtr.setTypeRC() fonksiyonu kullanılır.)
  qtr.setSensorPins((const uint8_t[]) {
    A0, A1, A2, A3, A4, A5, A6, A7
  }, 8);
  pinMode(AIN1, OUTPUT);
  pinMode(AIN2, OUTPUT);
  pinMode(BIN1, OUTPUT);
  pinMode(BIN2, OUTPUT);
  pinMode(LED, OUTPUT);
  pinMode(STBY, OUTPUT);

  delay(1000); //Araca enerji verince 1sn bekle
  kalibrasyon(1); // 0 elle, 1 otomatik kafa sallama.
}

void loop() {
  sensorOku();
  pd();
}

void sensorOku() {
  pozisyon = qtr.readLineWhite(sensor); // Beyaz çizginin pozisyonunu oku. (0 - 7000)
  hata = pozisyon - hedef; // Pozisyondan 3500 (hedef) çıkar. Hatayı bul.
  qtr.read(sensor); // Sekiz sensörün ham değelerini oku.
}

void pd() {
  turev = hata - sonHata; // Hatadan bir önceki hatayı çıkar.
  sonHata = hata; // Şimdiki hatayı kaydet.
 fark = ( hata * KP) + ( turev * KD ); // Motorlara uygulanacak farkı hesapla.

  constrain(fark, -maxHiz, maxHiz); // Fark en fazla maxHiz olsun.

  if ( fark < 0 ) // fark negatif ise
```

```
  motor(maxHiz, maxHiz + fark); // Sağ motorun hızını düşür.
  else // fark negatif değilse
    motor(maxHiz - fark, maxHiz); // Sol motorun hızını düşür.
}

void motor(int solMotorPWM, int sagMotorPWM) {
  digitalWrite(STBY, HIGH);

  if ( solMotorPWM >= 0 )  { // İleri.
    digitalWrite(BIN1, HIGH);
    digitalWrite(BIN2, LOW);
  }
  else  { // Negatifse geri döndür.
    digitalWrite(BIN1, LOW);
    digitalWrite(BIN2, HIGH);
    solMotorPWM *= -1;
  }
  analogWrite(PWMB, solMotorPWM);

  if ( sagMotorPWM >= 0 )  { // İleri.
    digitalWrite(AIN1, HIGH);
    digitalWrite(AIN2, LOW);
  }
  else  { // Negatifse geri döndür.
    digitalWrite(AIN1, LOW);
    digitalWrite(AIN2, HIGH);
    sagMotorPWM *= -1;
  }
  analogWrite(PWMA, sagMotorPWM);
}

void kalibrasyon(bool secim) { // 1 otomatik, 0 elle.
  if (secim) { // secim 1 ise otomatik kalibrasyon yap.
    byte hiz = 40; // Aracın kafasını sallama hızı.
    for (byte i = 0; i < 3; i++) {  // Sağa sola üç kez kafa salla.
      while (sensor[7] < 300) {
    motor(hiz, -hiz);
        qtr.calibrate();
        sensorOku();
      }
      while (sensor[7] > 700) {
        motor(hiz, -hiz);
        qtr.calibrate();
        sensorOku();
      }
```

```
      while (sensor[0] < 300) {
        motor(-hiz, hiz);
        qtr.calibrate();
        sensorOku();
      }
      while (sensor[0] > 700) {
        motor(-hiz, hiz);
        qtr.calibrate();
        sensorOku();
      }
      while (sensor[3] > 500)  { // Ortada dur.
        motor(hiz, -hiz);
        qtr.calibrate();
        sensorOku();
      }
    }
  } else { // Seçim 0 ise elle kalibrasyon yap.

    for ( byte i = 0; i < 70; i++)  { // Dahili LED yanıp söndüğü sürece (3 sn) elle kalibrasyon yap.
      digitalWrite(LED, HIGH); delay(20);
      qtr.calibrate();
      digitalWrite(LED, LOW); delay(20);
      qtr.calibrate();
    }
  }
  motor(0, 0);
  delay(2000); // Kalbirasyondan sonra 3 sn bekle.
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.98'deki devreyi araç üzerine kurunuz. Programı yükleyiniz.
4. Siyah zemin üzerine beyaz çizgiye aracı yerleştiriniz.
5. Araca güç verdikten sonra sekiz sensör siyah ve beyaz renkleri görecek şekilde üç saniye (dâhilî LED'in yanıp sönmesi bitene kadar) sensörü sola ve sağa gezdiriniz.
6. Araç çizgiden çıkıyorsa veya zikzak yapıyorsa KP ve KD değerlerini değiştiriniz.
7. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

1. Herhangi bir optik sensör kullanarak (QTR1C olabilir.) tekerlek üzerine işaretlenen beyaz veya siyah alanı (1, 2, 4 veya 8 nokta olabilir.) sayarak aracın aldığı yolu bulunuz.

```
// Aracın aldığı yolu cm cinsinden float veri tipinde döndüren fonksiyon
#define enkoderPini 2
bool yuzeyKontrol = 0;
int noktaSayisi = 0; // Tekerleğin üzerindeki yansıtıcı (beyaz) nokta sayısı.
float r = 1.55; // Tekerleğin yarıçapı (cm).

float enkoder() {

  if (digitalRead(enkoderPini) && yuzeyKontrol) { /* Optik sensör beyaz alanı gördüyse ve yuzey-
Kontrol değişkeni kurulu (1) ise.. */
    noktaSayisi++; // Sayacı bir arttır.

    yuzeyKontrol = beyaz; /* yuzeyKontrol bitini sıfırla (Beyaz alandan çıkmadığı sürece sayacı
arttırma-bu if yapısına girme). */

  }

  if (digitalRead(enkoderPini) == 0) // Optik sensör beyaz alandan çıkarsa...

    yuzeyKontrol = siyah; /* Sayım için yuzeyKontrol bitini kur (Sensör beyaz alandan çıktı). */


  float mesafe = 2 * 3.14 * r * noktaSayisi / 4; // cm cinsinden gidilen yol.

  /* 2*π*r (tekerleğin çevresi) * noktaSayisi/4 (tekerleğin üzerindeki yansıtıcı (beyaz) nokta
sayısı.) */
  return mesafe; // mesafe bilgisini döndür.
}
```

2. Yoldaki dik çizgilerin başlangıç noktasına olan uzaklıklarını EEPROM hafızaya kaydeden fonksiyonları yazınız.

```
void setup() { // setup fonksiyonuna eklenecek kodlar.
  Serial.begin(9600);

  // eepromReset(); Serial.println("Hafıza sıfırlanıyor. Lütfen bekleyiniz..."); // İhtiyaç du-
yulduğunda kullanılır.
  eepromOku();
}
bool dikCizgide = 0;
void sensorOku() { // Dik çizgiyi algılayan kodlar eklenir.


  pozisyon = qtr.readLineWhite(sensor); // Beyaz çizginin pozisyonunu oku. (0 - 7000)
  hata = pozisyon - hedef; // Pozisyondan 3500 (hedef) çıkar. Hatayı bul.
qtr.read(sensor); // Sekiz sensörün ham değerlerini oku.


  if (sensor[7] < 300 && sensor[6] < 300 && sensor[5] < 300 && sensor[4] < 300 && sensor[3] <
300 && sensor[2] < 300 && sensor[1] < 300 && sensor[0] < 300) // Dik çizgi varsa...
    dikCizgide = 1; // Dik çizginin içinde.
  else if (sensor[7] > 700 && sensor[0] > 700)
    dikCizgide = 0; // Dik çizgi geçildi.
```

```
}

// Dik çizgi sayısını döndüren fonksiyon. Ayrıca dik çizgilerin bulunduğu mesafeleri eeprom'a yazar.
int dikCizgiSayisi = 0;
bool dikCizgiKontrol = 1;

int dikCizgiSay() {

  if (dikCizgide && dikCizgiKontrol) { /* Optik sensörler beyaz alanı gördüyse ve yuzeyKontrol
değişkeni kurulu (1) ise.. */
    dikCizgiSayisi++; // Sayacı bir arttır.

    dikCizgiKontrol = beyaz; /* yuzeyKontrol bitini sıfırla (Beyaz alandan çıkmadığı sürece sa-
yacı arttırma-bu if yapısına girme). */
    EEPROM.write(dikCizgiSayisi - 1, enkoder()); /* Dik çizgileri  EEPROM hafızaya yaz. */
  }


  if (dikCizgide == 0) // Optik sensörler beyaz alandan çıkarsa...

    dikCizgiKontrol = siyah; /* Sayım için yuzeyKontrol bitini kur (Sensörler beyaz alandan çıktı). */
  return dikCizgiSayisi;
}
// enkoder değerlerini hafızadan okuma fonksiyonu
byte deger;

void eepromOku() {
  int hafizaBoyutu = EEPROM.length(); // Kartın hafıza boyutunu al.
  for (int adres = 0; adres < hafizaBoyutu; adres++) {
    deger = EEPROM.read(adres);

    if (deger) {
      Serial.print(adres + 1);
      Serial.print(". çizgi ");
      Serial.print(deger);
      Serial.println(" cm'dedir.");

  delay(1);
    }
  }
}
void eepromReset() {
  int hafizaBoyutu = EEPROM.length(); // Kartın hafıza boyutunu al.
  for (int adres = 0; adres < hafizaBoyutu; adres++) {
    EEPROM.write(adres, 0); // Tüm hafızayı sıfırla.
  }
}
```

3. Düzlüklerde aracı hızlandıran fonksiyonu yazınız.

```
// Girilen aralıklarda aracı hızlandıran fonksiyon.


void hizYap(float baslangic, float bitis, int virajMesafesi) { /* başlangıç noktası (cm), bitiş
noktası (cm), dönüşten önceki yavaşlama mesafesi (cm) */
  if (enkoder() > baslangic) {
    while (enkoder() < bitis) {
      sensorOku();
      pd();
      enkoder();
      if (enkoder() < bitis - virajMesafesi) {
        if (maxHiz < 255) { // Adım adım hızlan (en fazla 255).
          maxHiz++; // Hızlanma adımı.
          //delay(1); // Hızlanma gecikmesi.
        }
      } else maxHiz = 70; // Viraja girme hızı (en düşük hız).
    }
  }
}
```

4. Aracın önüne optik sensör (MZ80 vb.) ekleyerek sensörü engelde duracak şekilde düzenleyiniz.

5. Aracı L298N motor sürücüsüne göre düzenleyiniz.

```
// L298N motor sürücü pin tanımlamaları.
#define ENA 9 // A sağ motor.
#define IN1 8
#define IN2 7
#define IN3 5 // B sol motor.
#define IN4 4
#define ENB 3
```

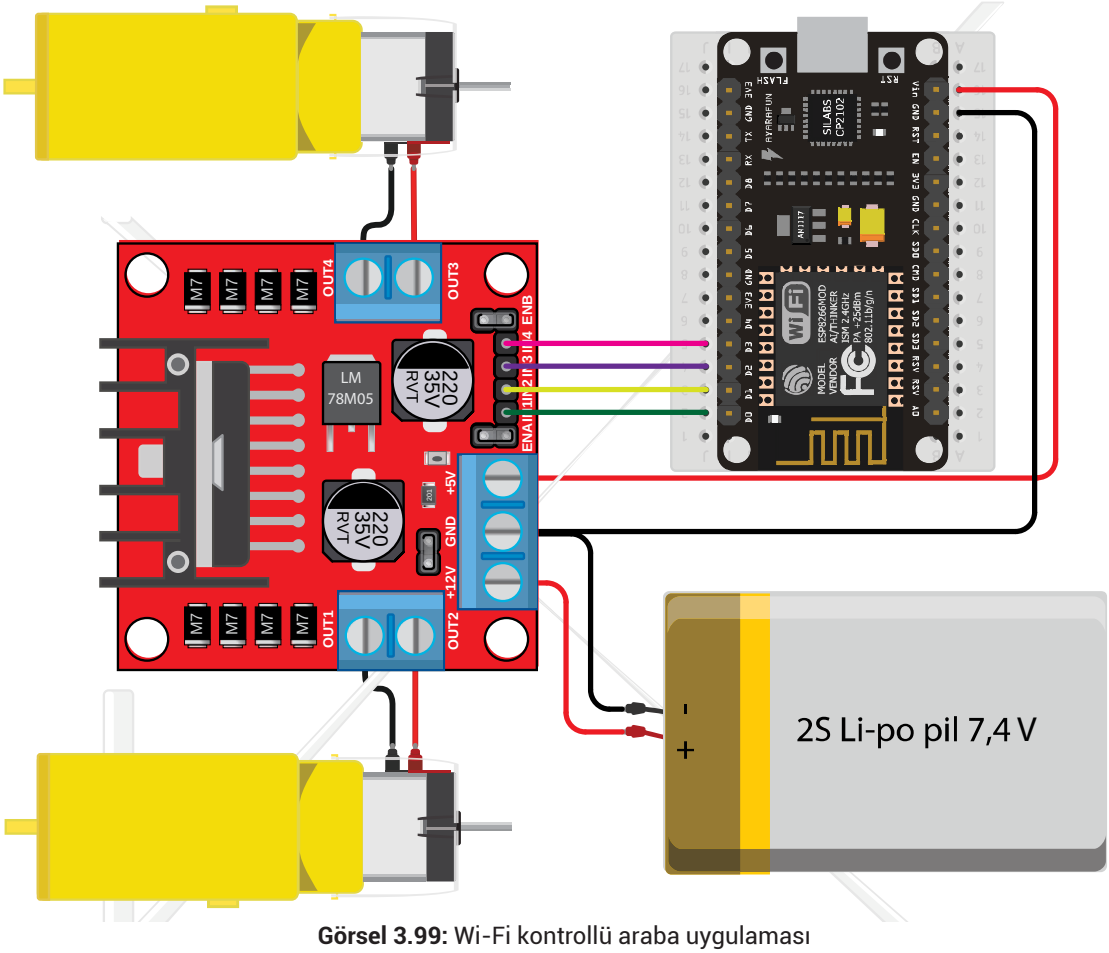
Sorular

1. Araç keskin dönüşler yapabilmekte midir? Neden?
2. **if (fark < 0) // fark negatif ise motor(maxHiz, maxHiz + fark); // Sağ motorun hızını düşür. else // fark negatif değilse motor(maxHiz - fark, maxHiz); // Sol motorun hızını düşür.** komut grubu yerine **motor(maxHiz - fark, maxHiz + fark);** komutu kullanılırsa aracın gidişinde nasıl bir değişim olur?
3. Aracın pozisyon bilgisi eş zamanlı olarak bluetooth üzerinden cep telefonuna aktarılabilir mi? Araştırınız.

Uygulama Adı: Wi-Fi Kontrollü Araba Uygulaması No.: 20

Amaç: Wi-Fi kontrollü araba uygulaması yapmak.

Görsel 3.99'da Wi-Fi kontrollü araba uygulamasında NodeMCU acces point (AP) modunda çalışmaktadır. NodeMCU'ya bağlanmak için kablosuz ağlardan "Robot" seçilir. Şifre "12345678" girilir. Bilgisayar veya akıllı telefondan tarayıcı (Chrome vb.) açılarak adres satırına 192.168.4.1 girilerek NodeMCU içindeki web server sayfasından araç kontrol edilir (Görsel 3.100).



Görsel 3.99: Wi-Fi kontrollü araba uygulaması



Görsel 3.100: Tarayıcıdan NodeMCU'ya bağlanma

Wi-Fi kontrollü araba programı aşağıdaki gibidir:

```
/*  NodeMCU Access Point (AP) modunda çalışmaktadır.
    Kablosuz ağlardan «Robot»a bağlanın. Şifre: 12345678
    Tarayıcıdan 192.168.4.1 adresine girin.
*/
#include <ESP8266WiFi.h> // NodeMCU kurulumuyla gelir.
#include <WiFiClient.h> // NodeMCU kurulumuyla gelir.
#include <ESP8266WebServer.h> // NodeMCU kurulumuyla gelir.
#include <WebSocketsServer.h> // WebSockets v2.3.6
#include <Hash.h> // NodeMCU kurulumuyla gelir.

const char* ssid = "Robot";
const char* password = "12345678"; // Şifre en az sekiz haneli olmalı.

const byte IN1 = D0, IN2 = D1, IN3 = D2, IN4 = D3; // PWM pinleri. (enA ve enB'de jumper takılı).

int hiz = 150;
```

```
char karakter;

static const char PROGMEM INDEX_HTML[] = R"rawliteral(
<!DOCTYPE html>
<html>
<head>
<meta name = "viewport" content = "width = device-width, initial-scale = 1.0, maximum-scale = 1.0, user-scalable=0">
<title>Wifi Robot</title>
<style>
"body { background-color: #808080; font-family: Arial, Helvetica, Sans-Serif; Color: #000000; }"
#JD {
  text-align: center;
}
#JD {
  text-align: center;
  font-family: "Lucida Sans Unicode", "Lucida Grande", sans-serif;
  font-size: 24px;
}
.foot {
  text-align: center;
  font-family: "Comic Sans MS", cursive;
  font-size: 30px;
  color: #F00;
}
.button {
    border: none;
    color: white;
    padding: 17px;
    text-align: center;
    text-decoration: none;
    display: inline-block;
    font-size: 16px;
    margin: 4px 2px;
    cursor: pointer;
    border-radius: 12px;
  width: 100%;
}
.red {background-color: #F00;}
.green {background-color: #090;}
.yellow {background-color:#F90;}
.blue {background-color:#03C;}
</style>
<script>
```

```
var websock;
function start() {
  websock = new WebSocket('ws://' + window.location.hostname + ':81/');
  websock.onopen = function(evt) { console.log('websock open'); };
  websock.onclose = function(evt) { console.log('websock close'); };
  websock.onerror = function(evt) { console.log(evt); };
  websock.onmessage = function(evt) {
    console.log(evt);
    var e = document.getElementById('ledstatus');
    if (evt.karakter === 'ledon') {
      e.style.color = 'red';
    }
    else if (evt.karakter === 'ledoff') {
      e.style.color = 'black';
    }
    else {
      console.log('unknown event');
    }
  };
}
function buttonclick(e) {
  websock.send(e.id);
}
</script>
</head>
<body onload="javascript:start();">
 
<table width="100%" border="0">
  <tr>
    <td bgcolor="#FFFFFF" id="JD">Wifi Robot </td>
  </tr>
</table>
<table width="100" height="249" border="0" align="center">
    <td> </td>
    <td align="center" valign="middle"><form name="form1" method="post" action="">
      <label>

        <button id="a"  type="button" onclick="buttonclick(this);" class="button green">ileri</button>
      </label>
    </form></td>
    <td> </td>
  </tr>
    <td align="center" valign="middle"><form name="form1" method="post" action="">
      <label>
        <button id="c"  type="button" onclick="buttonclick(this);" class="button green">Sol  </
button>
```

```
var websock;
function start() {
  websock = new WebSocket('ws://' + window.location.hostname + ':81/');
  websock.onopen = function(evt) { console.log('websock open'); };
  websock.onclose = function(evt) { console.log('websock close'); };
  websock.onerror = function(evt) { console.log(evt); };
  websock.onmessage = function(evt) {
    console.log(evt);
    var e = document.getElementById('ledstatus');
    if (evt.karakter === 'ledon') {
      e.style.color = 'red';
    }
    else if (evt.karakter === 'ledoff') {
      e.style.color = 'black';
    }
    else {
      console.log('unknown event');
    }
  };
}
function buttonclick(e) {
  websock.send(e.id);
}
</script>
</head>
<body onload="javascript:start();">
 
<table width="100%" border="0">
  <tr>
    <td bgcolor="#FFFFFF" id="JD">Wifi Robot </td>
  </tr>
</table>
<table width="100" height="249" border="0" align="center">
    <td> </td>
    <td align="center" valign="middle"><form name="form1" method="post" action="">
      <label>
        <button id="a"  type="button" onclick="buttonclick(this);" class="button green">ileri</button>
      </label>
    </form></td>
    <td> </td>
  </tr>
    <td align="center" valign="middle"><form name="form1" method="post" action="">
      <label>
        <button id="c"  type="button" onclick="buttonclick(this);" class="button green">Sol  </
button>
```

```
      </label>
    </form></td>
  <td align=”center” valign=”middle”><form name=”form1” method=”post” action=””>
      <label>
        <button id=”e”  type=”button” onclick=”buttonclick(this);” class=”button red”>Dur  </
button>
      </label>
    </form></td>
    <td align=”center” valign=”middle”><form name=”form1” method=”post” action=””>
      <label>
        <button id=”d”  type=”button” onclick=”buttonclick(this);” class=”button green”>Sag  </
button>
      </label>
    </form></td>
  </tr>
   <td> </td>
    <td align=”center” valign=”middle”><form name=”form1” method=”post” action=””>
      <label>
        <button id=”b”  type=”button” onclick=”buttonclick(this);” class=”button green”>Geri </
button>
      </label>
    </form></td>
    <td> </td>
  </tr>
</table>
</body>
</html>
)rawliteral”;

WebSocketsServer webSocket = WebSocketsServer(81);
ESP8266WebServer server(80);

void setup() {
 Serial.begin(9600);

 pinMode(IN1, OUTPUT);
 pinMode(IN2, OUTPUT);
 pinMode(IN3, OUTPUT);
 pinMode(IN4, OUTPUT);

  WiFi.mode(WIFI_AP);           // Access point modunda.
 WiFi.softAP(ssid, password);  // Kablosuz ağı başlat.
```

```
  IPAddress IP = WiFi.softAPIP();
 server.on("/", [](){
  server.send(200, "text/html", INDEX_HTML);
  });

  server.begin();
  webSocket.begin();
  webSocket.onEvent(webSocketFonksiyonu);
}

void loop() {

if (karakter == 'a') { // ileri.
      analogWrite(IN1, hiz);
      analogWrite(IN2, 0);
      analogWrite(IN3, hiz);
      analogWrite(IN4, 0);
    } else if (karakter == 'b') { // Geri.
      analogWrite(IN1, 0);
      analogWrite(IN2, hiz);
      analogWrite(IN3, 0);
      analogWrite(IN4, hiz);
    } else if (karakter == 'c') { // Sol.
      analogWrite(IN1, hiz);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, hiz);
    } else if (karakter == 'd') { // Sağ.
      analogWrite(IN1, 0);
      analogWrite(IN2, hiz);
      analogWrite(IN3, hiz);
      analogWrite(IN4, 0);
    } else if (karakter == 'e') { // Dur.
      analogWrite(IN1, 0);
      analogWrite(IN2, 0);
      analogWrite(IN3, 0);
      analogWrite(IN4, 0);
    }

   webSocket.loop();
   server.handleClient();
}

void webSocketFonksiyonu(byte sayi, WStype_t type,byte * veri, size_t length){
  switch(type) {
```

```
    case WStype_DISCONNECTED:
      break;
    case WStype_CONNECTED:
      {IPAddress ip = webSocket.remoteIP(sayi);}
      break;
    case WStype_TEXT:
      karakter = veri[0];
      Serial.println(karakter);
      webSocket.broadcastTXT(veri, length);
      break;
    case WStype_BIN:
      hexdump(veri, length);
      webSocket.sendBIN(sayi, veri, length);
      break;
      default:
      break;
  }
}
```

İşlem Basamakları

1. İş sağlığı ve güvenliği tedbirlerine uyarak araç gerecinizi hazırlayınız.
2. Öğretmeninizden teslim aldığınız malzemelerin gerekli kontrollerini yapınız.
3. Görsel 3.99'daki devreyi araç üzerine kurunuz. Programı yükleyiniz.
4. Bilgisayar veya akıllı telefondan "Robot" isimli kablosuz ağa "12345678" şifresiyle bağlantı gerçekleştiriniz.
5. Tarayıcı adres çubuğuna "192.168.4.1" yazınız.
6. Web sayfası üzerindeki düğmelerle aracı yönlendiriniz.
7. Öğretmeninizden teslim aldığınız malzemeleri gerekli kontrollerini yaparak iade ediniz.

SIRA SİZDE

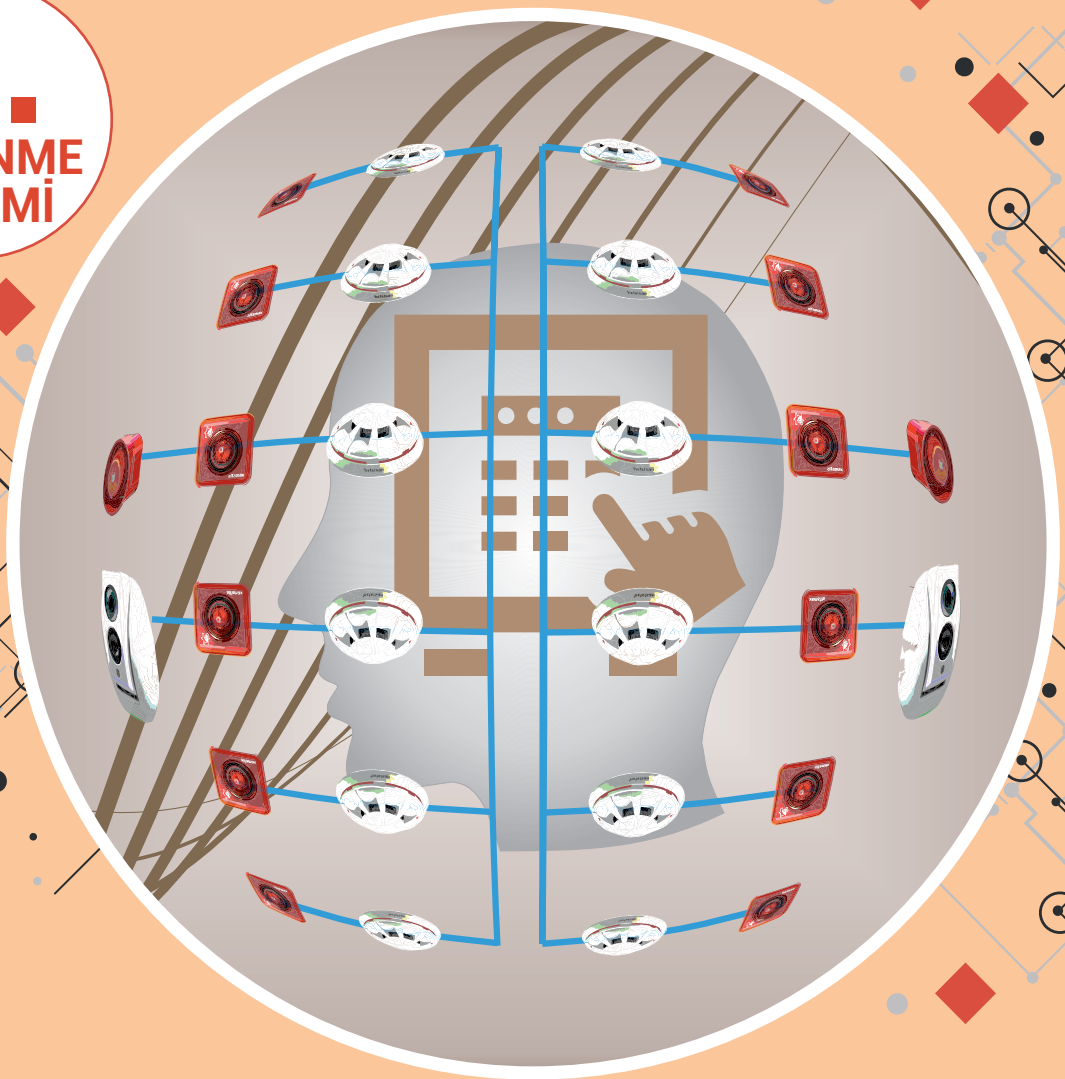
1. Araç korna çalacak şekilde donanım ve yazılımı düzenleyiniz.

Soru

1. NodeMCU üzerinde çalışan **web server**a kaç adet istemci bağlanabilir? Araştırınız.

NOT DEFTERİ

4.
ÖĞRENME
BİRİMİ



YANGIN ALARM SİSTEMLERİ

NELER ÖĞRENECEKSİNİZ ?

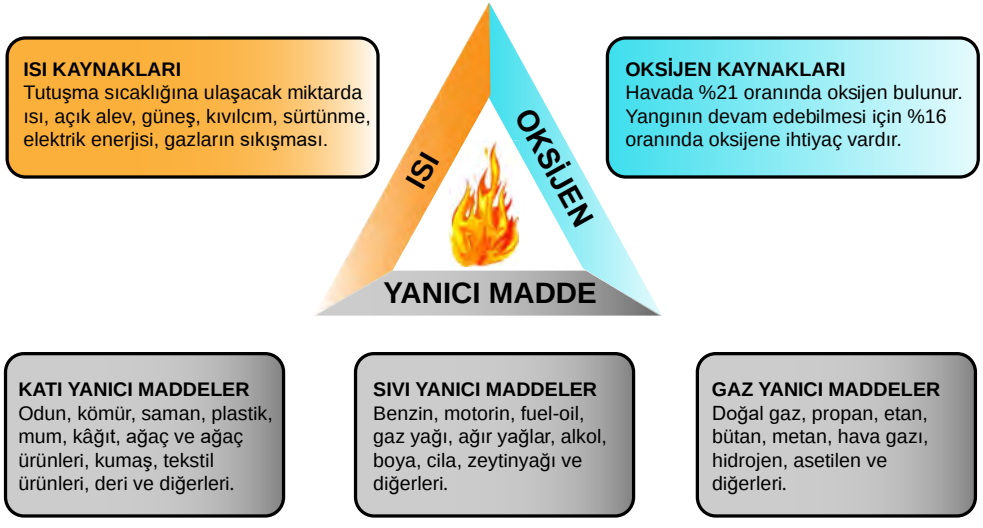
- Yangın Alarm Sistemleri
- Yangın Alarm Santrali
- Yangın Alarm Sistemi Donanımları

TEMEL KAVRAMLAR

yangın
karbonmonoksit
ısı
ışık
alev

4. YANGIN ALARM SİSTEMLERİ

Yanıcı maddenin tutuşma sıcaklığında yeterli oranda oksijen ile buluşması sonucu meydana gelen kimyasal reaksiyona **yanma** denir. Yanma olayının gerçekleşebilmesi için üç unsurun uygun oranlarda bir araya gelmesi gerekir. Bunlar; oksijen, ısı (sıcaklık) ve yanıcı maddedir. Bu üç unsurun bir araya gelmesine aynı zamanda **yanma üçgeni** de denir (Görsel 4.1).



Görsel 4.1: Yanma üçgeni

Yangın sonucunda duman, gaz (karbonmonoksit ve karbondioksit), alev, ısı (sıcaklık) ve ışık gibi unsurlar ortaya çıkar. Yangın alarm sistemleri; yanma sonucu ortaya çıkan bu ürünlerin fark edilmesi, buna bağlı olarak çıkışında sesli ve ışıklı uyarı ile kontrol cihazlarını çalıştırması esasına göre tasarlanır ve üretilir.

Başka bir ifade ile yangın alarm sistemleri, yangını henüz başlangıç aşamasındayken algılayan ve ihbar cihazları aracılığı ile insanları uyaran sistemlerdir. Yangın alarm sistemlerinin asıl amacı yangın çıktığı andan itibaren oluşabilecek can ve mal kayıplarını olabildiği kadar engellemektir.

Yangın alarm sistemleri, evler, ofisler, fabrikalar, alışveriş merkezleri, okullar, oteller, öğrenci yurtları, hastaneler vb. tüm kapalı alanlarda kullanılmaktadır.

Yangın alarm sistemlerinde kullanılan ürünler şunlardır (Görsel 4.2):

1.Yangın Alarm Santrali: Yangın alarm sisteminin beyni yani kontrol merkezidir. Dedektörlerden veya yangın ihbar butonlarından alınan elektriksel sinyaller burada işlenerek siren, GSM modülü, röle, ikaz LED'leri gibi çıkış elemanlarının çalıştırılmasını sağlar.

2.Dedektörler: Isı, duman, gaz ve alev gibi yangın unsurlarını sensörleri vasıtasıyla algılayarak bilgileri elektriksel sinyal biçiminde yangın alarm santraline aktaran ürünlerdir. Dedektörlerden gelen bilgiye göre yangın alarm santrali çalışma durumunu belirler.

3.Yangın İhbar Butonları: Yangının henüz başlangıç aşamasında veya yangın olayının fark edildiği durumda yangın alarm santraline yangın alarmı vermesi için gerekli bilgiyi göndermek için kullanılır.

4.Sirenler: Yangın alarm santrali, dedektörlerden veya yangın ihbar butonlarından "Yangın var." bilgisini aldıktan sonra flaşörlü veya flaşörsüz sireni aktif eder. Yüksek ses şiddeti ve fasılalı ışık sinyali ile ortamın tehlikeli olduğu haberini verir.

5.GSM Modülü: Yangın alarm santrali "Yangın var." bilgisini aldıktan sonra GSM modülünü aktif eder. Hafızasına kayıtlı olan numaraları arar veya SMS göndererek kontrol edilen ortamda yangın algılandığını bildirir.

6.Acil Çıkış Yönlendirme Levhaları: Yangın alarm santralinde çıkan yangın anında insanlara paniğe kapılmadan en kısa yoldan ortamdan dışarı çıkış yolunu gösteren yönlendirme levhalarıdır.

7.Söndürme Sistemleri: Yangın alarm santralinden gelen bilgiye göre yangını söndürmek için çalışan sisteme **otomatik yangın söndürme sistemi** denir. İster otomatik sistemler ister manuel yangın tüpleri olsun hepsinin amacı ortamda çıkan yangını söndürmektir.

8.Yangın Kabloları: Yangın alarm sistemlerinde kullanılan kabloların yangına en az 60 dakika dayanıklı olmaları gerekir. Yangın alarm santrali ile diğer ekipmanlar arasında kullanılan kablolara ek yapılmamalıdır. Ek yapılmak zorunda kalındığında ise yangına dayanıklı özel ekipmanlar tercih edilmelidir.



Görsel 4.2: Yangın alarm sistemlerinde kullanılan ürünler

4.1. YANGIN ALARM SANTRALİ

Yangın alarm santrali ile bina içinde 7/24 sürekli olarak kontrol sağlanmaktadır. Yangın alarm santrali; herhangi bir nedenle oluşmaya başlamış bir yangını sezip gereken önlemlerin alınması, alanın boşaltılması ve söndürme işlemlerinin yapılabilmesi için belli yerlere komut veren, programlanabilir bir kontrol sistemidir.

Yangın alarm santrallerinin genel çalışma ilkesi, dedektörlerden veya yangın ihbar butonlarından gelen sinyallere göre çıkışında sesli ve ışıklı uyarı cihazlarını, telefon arayıcı ürünlerini veya internetten haber alma merkezlerini harekete geçirmesi esasına dayanır. Ayrıca sisteme bağlı olması hâlinde otomatik

söndürme sistemini de harekete geçirir. Görsel 4.3'te genel bir yangın alarm santralinin prensip şeması görülmektedir.



Görsel 4.3: Yangın alarm santralinin prensip şeması

Yangın alarm santrallerinin çalışma modları

Denge modunda: Kontrol edilen bölgenin (**zone**) dedektör ve yangın ihbar butonlarının seviyesinde herhangi bir değişme yok demektir. Bu durum yangın alarm santralinde bir LED uyarıcı ile belirtilir.

Hata modunda: Dedektörlerde, yangın ihbar butonlarında, iletim hatlarında açık devre hatası veya kontrol panelinde bir arıza (kısa devre) var demektir. Bu durum, yangın alarm santralinde sarı LED'lerin yanması ve kesik kesik sesli uyarıcı sinyaller ile belirtilir.

Alarm modunda: Kontrol edilen bölgelerin birinde veya birden fazlasında yangın çıkmış demektir. Bu durumda yangın alarm santrali, sesli uyarı verebilmek için sireni, ışıklı uyarı verebilmek için ise paneldeki kırmızı LED'leri kullanır. Aynı zamanda GSM modülü veya internet ile uzaktan kontrol eden haber alma merkezlerine gerekli uyarıları yapar.

Yangın alarm santralleri, **konvansiyonel sistem** ve **adresli sistem** olmak üzere iki çeşittir. Konvansiyonel sistemlerde bina belirli bölgelere (**zone**) ayrılarak her bölgenin özelliğine göre oraya belirli sayıda dedektör takılır. Yangın sırasında algılama ile panelde dedektörün ait olduğu bölge numarası ile alarm verilir fakat hangi dedektörden alarm geldiği tespit edilememektedir.

Adreslenebilir sistemlerde ise nokta bazında algılama yapılmaktadır. Yangın sırasında ihbarın hangi dedektörden geldiği tespit edildiğinden yangının erken fark edilmesi sağlanmaktadır. Örneğin 4 katlı bir binada kurulu olan sistemde yangın algılandığında konvansiyonel sistemde "3. katta yangın var." ikazı verilirken adreslenebilir sistemde "3. katta 3. odada yangın var." ikazı verilir.

4.1.1. Konvansiyonel Sistem Yangın Alarm Santrali

Görsel 4.4'te farklı modellerdeki konvansiyonel sistem yangın alarm santralleri görülmektedir.



Görsel 4.4: Farklı modellerdeki konvansiyonel sistem yangın alarm santralleri

Konvansiyonel yangın alarm santrali, çıkışında kontrol edeceği **zone** (bölge) sayısına göre isimlendirilir. Her **zone**a genel olarak 20 dedektör ve istenen sayıda yangın ihbar butonu bağlanabilir. Bunun için konvansiyonel yangın alarm santrali, ufak ve orta ölçekli ortamlar için tercih edilir.

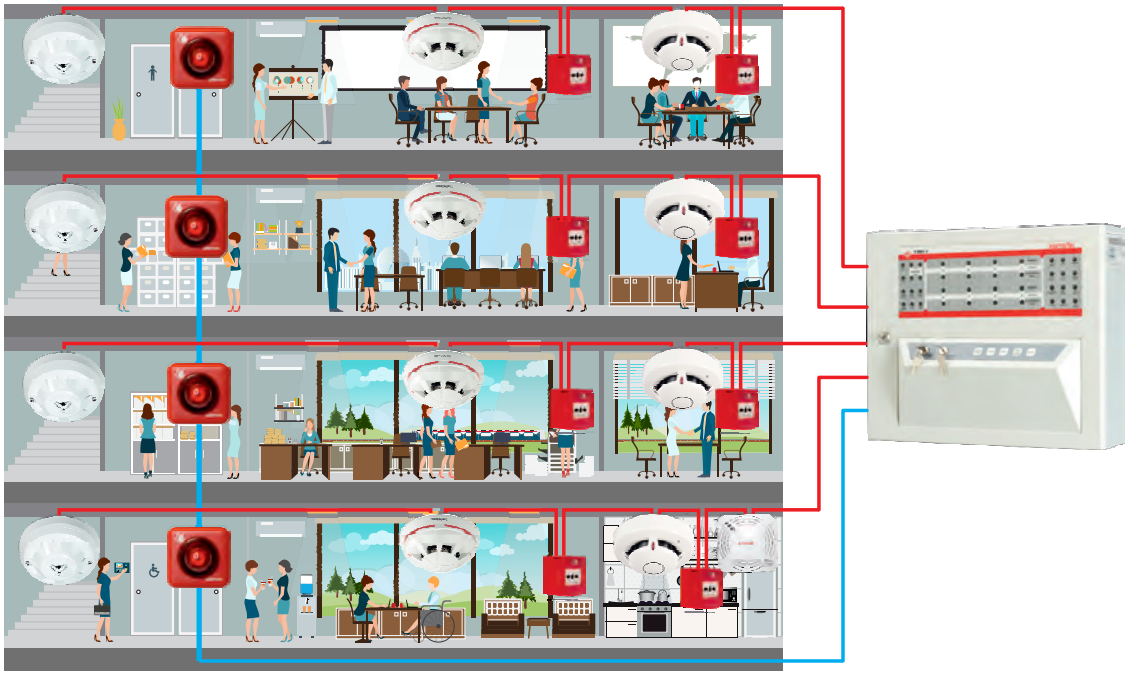
Konvansiyonel yangın alarm santrali seçilirken montaj yapılacak mimari yapının özelliğine göre seçim yapılmalıdır. Yangın alarm santrali alırken kullanıcıların genel hatası ise toplam dedektör sayısını 20'ye bölüp "Nasılsa bir bölgeye 20'ye kadar dedektör bağlanabiliyor." mantığı ile kapasite seçimi yapmalarıdır.

Bir **zone**a bağlanacak dedektör sayısını, seçilen bölgenin mimari yapısı belirlemelidir. Örnek olarak 4 katlı bir binanın her katında 4 dedektör varsa 4 katın tamamını bir bölge yapıp yangın ihbar panelinin bir bölgesine bu 4 katı bağlamak son derece yanlış bir uygulamadır. Her kat ayrı ve en az 1 bölge olarak tanımlanmalıdır.

Önemli olan husus, yangın ihbarı verildiğinde ilgili bölgeye en kısa zamanda ulaşıp yangına müdahale etmektir. Bölgelendirme yanlış yapılmışsa örneğin 2 kat 1 bölge olarak tanımlanmışsa herhangi bir yangın durumunda müdahale edecek kişilerin iki kata birden bakması gerekecektir. Bu da yangına daha geç ulaşılmasına dolayısıyla can ve mal kaybına sebep olabilecektir.

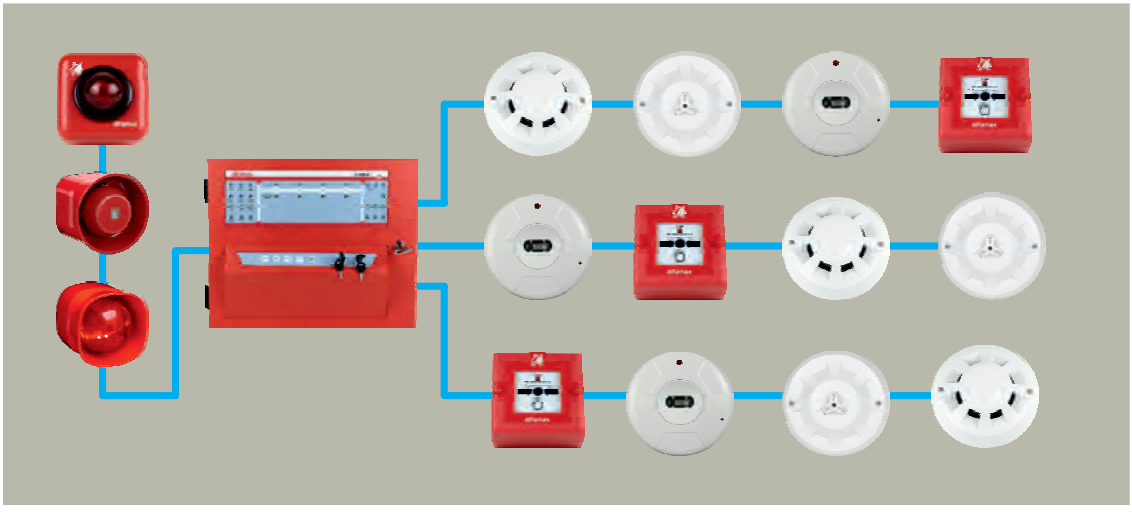
Herhangi bir dedektör yangın algıladığında veya yangın ihbar butonuna basıldığında bahse konu olan dedektör, panel üzerinde hangi **bölge**ye ait ise o bölgeden uyarı sinyali tetiklemektedir. Konvansiyonel sistemlerde, **bölge**ler üzerinde bulunan hangi dedektörün sinyal verdiğinin bilinmesi mümkün değildir. Sadece o elemanın ait olduğu bölgenin numarası ve adı görüntülenebilmektedir.

Görsel 4.5'te konvansiyonel bir yangın alarm sisteminin 4 katlı bir binaya yerleşimi ve yangın alarm santrali ile bağlantısı görülmektedir.



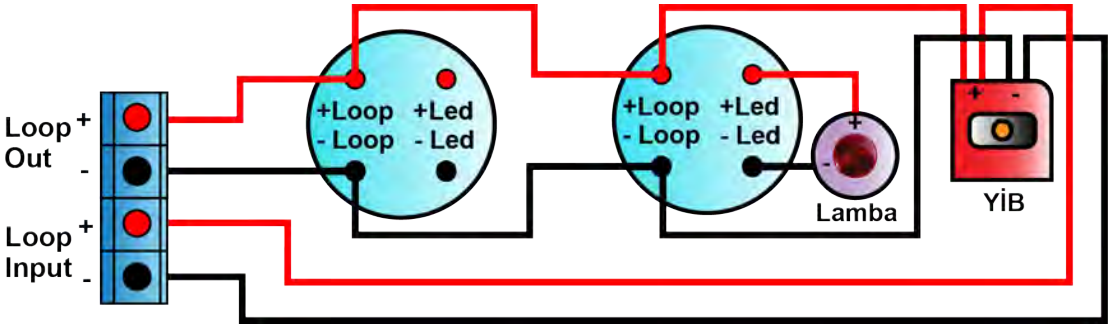
Görsel 4.5: 4 katlı bir binada konvansiyonel yangın algılama ve alarm sistemi

Görsel 4.6'da 4 bölgeli konvansiyonel yangın alarm sisteminin, yangın alarm santrali ile bölgelere bağlanacak dedektör ve yangın ihbar butonlarının bağlantısı görülmektedir. Siren çıkışına bağlanan sirenler paralel bağlanmıştır.

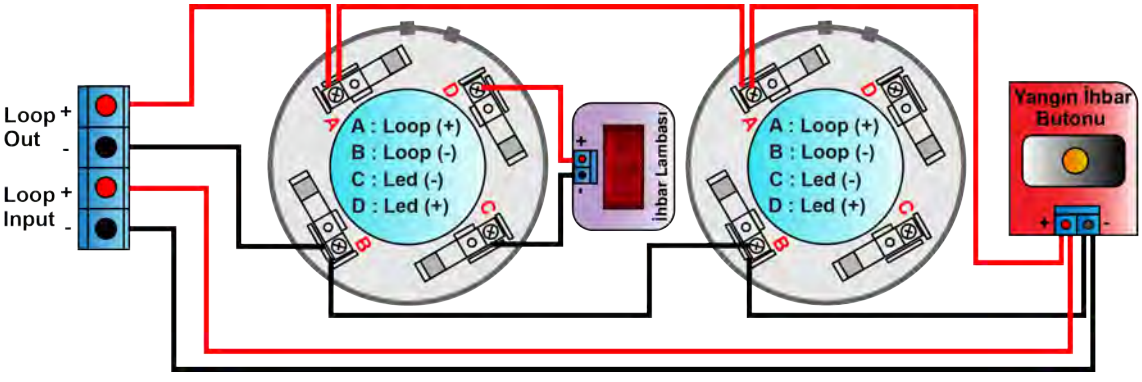


Görsel 4.6: 4 bölgeli konvansiyonel yangın algılama ve alarm sisteminin bağlantısı

Konvansiyonel yangın alarm santralinin temel bağlantı şeması Görsel 4.7a'da görülmektedir. Bölgelere dedektörler bağlanırken ikili yanmaz kablo üzerinden enerji gönderilir. Sistem, kırmızı (+,artı) kablonun dedektörlerin kontakları üzerinden birbirine seri bağlantısı yapılarak çalışır. Siyah (-,eksi) kablo ise dedektörlere paralel olarak girer ve çıkar. Görsel 4.7b'de ise konvansiyonel yangın alarm santralinin temel montaj şeması görülmektedir.



Görsel 4.7a: Konvansiyonel yangın alarm santralinin temel bağlantı şeması



Görsel 4.7b: Konvansiyonel yangın alarm santralinin temel montaj şeması

Dedektör bağlantıları, alt tabandaki bağlantı pinlerinin isimlerine dikkat edilerek yapılır. Modeller arasında bağlantı pinlerinin numaraları farklı olsa da bağlantı mantıkları aynıdır. Yangın alarm santrali üzerindeki **Z+** ucuna bağlanan kırmızı kablo, dedektörün tabanında yazan **+IN** (IN+ veya +Giriş) ucuna bağlanır. **Z-** ucu ise siyah kablo ile dedektördeki **-IN/OUT** (-COM veya -Ortak) uçlarına bağlanır. Böylece birinci dedektörün yangın alarm paneline bağlantısı tamamlanır. İkinci dedektör bağlantısı için birinci dedektörün kırmızı kablo ile **+OUT** (+Çıkış) ucundan ikinci dedektörün+IN ucuna bağlantı yapılır. Böylece kırmızı (artı,+) kablo ile iki dedektör seri bağlanmış olur. **–IN/OUT** uçları birbirine siyah kablo ile bağlanır. Siyah kablo ile **–IN/OUT** uçları paralel bağlanmış olur. Üçüncü dedektör bağlantısı için ikinci dedektörün kırmızı kablo ile **+OUT** ucundan üçüncü dedektörün +IN ucuna bağlantı yapılır. **–IN/OUT** uçları birbirine siyah kablo ile bağlanır. **+OUT ucu ile –IN/OUT** uçları arasına hat sonu direnci (4K7, 6K8 değerinde) bağlantısı yapılarak 3 dedektör bağlantısının yangın alarm panelindeki ilgili **zone** bağlantısı yapılmış olur.

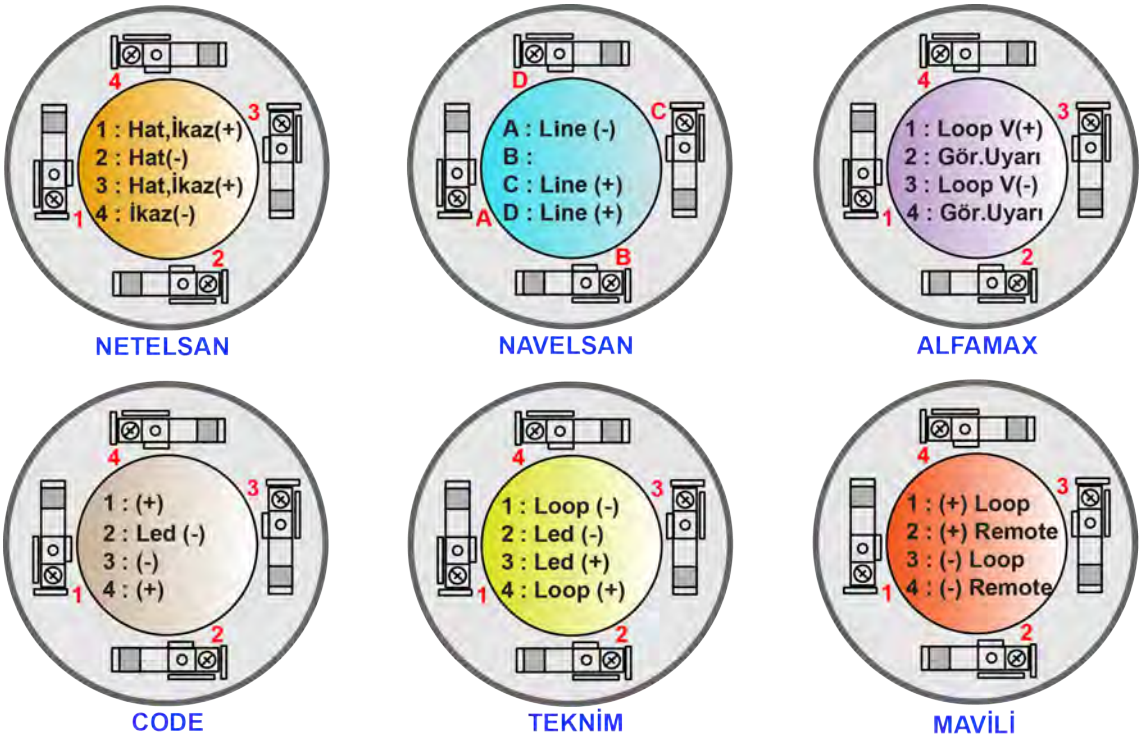
Dedektörlerin sonuna mutlaka hat sonu direnci eklenmelidir. Çünkü sistemin denge modunda çalışmasında sistemin geri dönüşü hat sonu direnci üzerinden yapılır. Kısaca hat sonu direncinden geçen akıma göre konvansiyonel yangın alarm santrali, hata moduna, denge moduna veya alarm moduna geçer.

Sistem denge modunda çalışırken hat sonu direncinden (4K7, 6K8) geçen akımı örnek almaktadır. **Zone**da bir temassızlık veya kopukluk olma durumunda **zone** hattı açık devre olacağı için **zone**dan akım geçişi olmaz ve yangın alarm santrali hata modunda çalışır. Yine **zone** hattında bir kısa devre olduğunda **zone**dan aşırı akım çekileceği için yangın alarm santrali yine hata modunda çalışır ve santral ilgili **zone**u devre dışına alır. Yangın alarm santralin alarm modunda çalışması için hattın direncinin dedektör, yangın ihbar butonları veya diğer dedektörler (doğal gaz, LPG gibi) aracılığı ile 470 Ω'a (680 Ω) düşürülmesi gerekir.

Zone hattı 470 Ω'a düştüğünde yangın alarm santrali ilgili **zone**dan çekilen alarm akımını algılar ve yangın alarm santrali alarm modunda çalışır.

Yangın ihbar butonları **zone** uçlarına **+IN** veya **+OUT** ile **–IN/OUT** arasına paralel olarak bağlanır. LED (tetik, ikaz) ucu ise ihbar lambasını bağlamak veya dışarıdan röle üzerinden farklı cihazları kontrol etmek için kullanılır.

Görsel 4.8'de farklı modellerdeki dedektörlerin tabanlarındaki bağlantı uçlarının görselleri görülmektedir. İşlevlerine göre bağlantı uçları farklı olsa bile bağlantı mantığı hepsinde aynıdır.



Görsel 4.8: Farklı konvansiyonel dedektörlerin tabanlarındaki bağlantı uçlarının isimleri

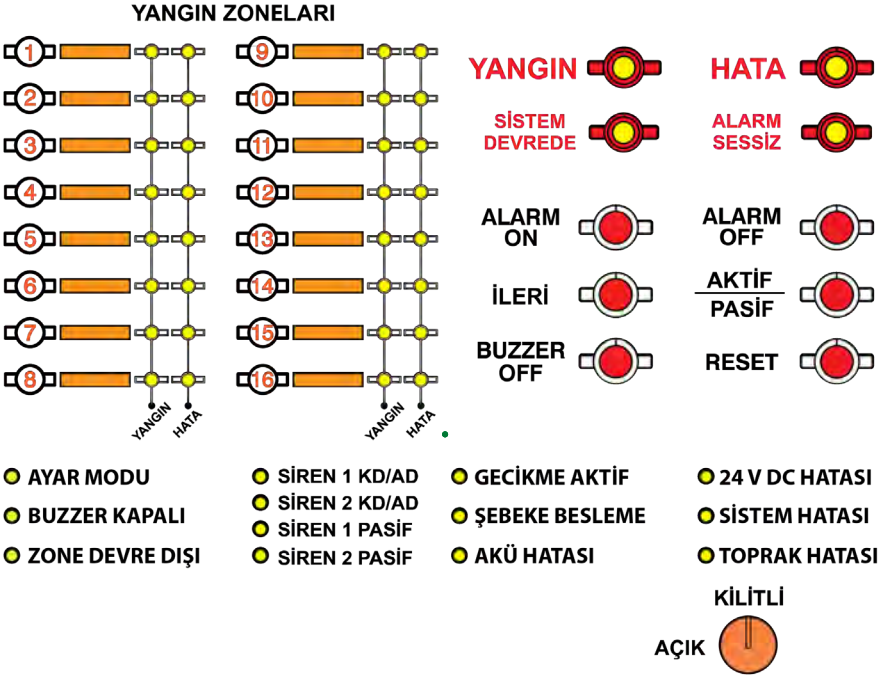
Tercih edilen ürünün konvansiyonel sisteminde, kullanılan dedektörlerin hepsinin montaj ayakları ve gövdelerinin dış plastik kasa büyüklükleri aynıdır. Dedektörlerin taban ayaklarında 4 adet bağlantı yapılabilir klemens vardır. Dedektörler ile yangın alarm sistemi arasındaki bağlantılar yapılırken bu klemensler kullanılır. Klemens numaraları ve bağlantı şemasındaki numaralandırmalar tercih edilen ürüne göre farklılık gösterir. Dedektör taban ayağındaki numaraların ilgili bağlantı şemasına göre hangisi kullanılır ise kullanılsın sistemin prensip şeması Görsel 4.7a'daki gibidir. Numaralandırma sadece model farkını belirtir. Sistemi kurarken sorun yaşamamak için tercih edilen modelin ürünleri kullanılmalıdır.

Görsel 4.9'da konvansiyonel sistem yangın alarm santralinin örnek ön yüzü görülmektedir. Konvansiyonel sistem yangın alarm santralinin üzerindeki göstergeler, butonlar ve işlevleri genel olarak aşağıda verilmiştir. Fakat günümüzde konvansiyonel yangın alarm sistemlerinde grafik ekran, touchscreen ekran ve OLED ekranların gelişimi ile zamanla bu göstergeler yerini yeni yüz ekranlara bırakacaktır.

Göstergeler

- **Sistem devrede LED'i,** santrale enerji geldiği sürece sürekli yanar.

. **Hata LED**'i santraldeki bütün hata durumlarında yanar, hatanın ortadan kalkmasıyla söner.
. **Zone hata LED'leri,** zone hattında kısa devre veya açık devre olması, hat sonu direnci olmaması veya herhangi bir dedektörün çıkarılması durumunda yanar.
. **Yangın LED'i,** herhangi bir **zone**da yangın alarm durumu oluşması durumunda yanar, ana kart üzerindeki **buzzer** ve sirenler sürekli çalar ve yangın alarm rölesi çıkışı aktif olur.
. **Zone yangın LED'leri** yangın durumunun oluşması durumunda yanar ancak sistemin resetlenmesi ile söner.
. **Program girişi LED'i;** test, devre dışı veya gecikme süresi verme işlemleri sırasında yanar.
. **Güç hatası LED'i,** santralde şebeke gerilimi bulunmadığı zaman yanar, şebeke gerilimi geldiğinde söner.
. **Devre dışı LED'i,** herhangi bir **zone**un devre dışı bırakılması durumunda yanar.
. **Test LED'i, zone**un test edilmesi sırasında yanar.
. **Siren 1 A/D LED'i,** siren 1 hattında kopuk olduğu zaman yanar.
. **Siren 1 K/D LED'i,** siren 1 hattında kısa devre olduğu zaman yanar.
. **Siren 2 A/D LED'i,** siren 2 hattında kopuk olduğu zaman yanar.
. **Siren 2 K/D LED'i,** siren 2 hattında kısa devre olduğu zaman yanar.
. **Akü hatası LED'i,** akü bağlantısı olmadığında veya akü voltajı18 V'un altına düştüğünde yanar.
. **Gecikme aktif LED'i,** siren çıkışlarına gecikme verilmesi durumunda yanar.
. **24 V DC hatası LED'i,** yardımcı 24 V çıkışından fazla akım çekilmesi durumunda yanar.
. **İşlemci hatası LED'i,** ana kart üzerindeki işlemcinin çalışmadığı durumda yanar.
. **Toprak hatası LED'i,** toprak hattında voltaj kaçağı olması durumunda yanar.
. **Alarm iptal LED'i,** sirenlerin susturulması durumunda yanar.



Görsel 4.9: Konvansiyonel sistem yangın alarm santralinin örnek ön yüzü

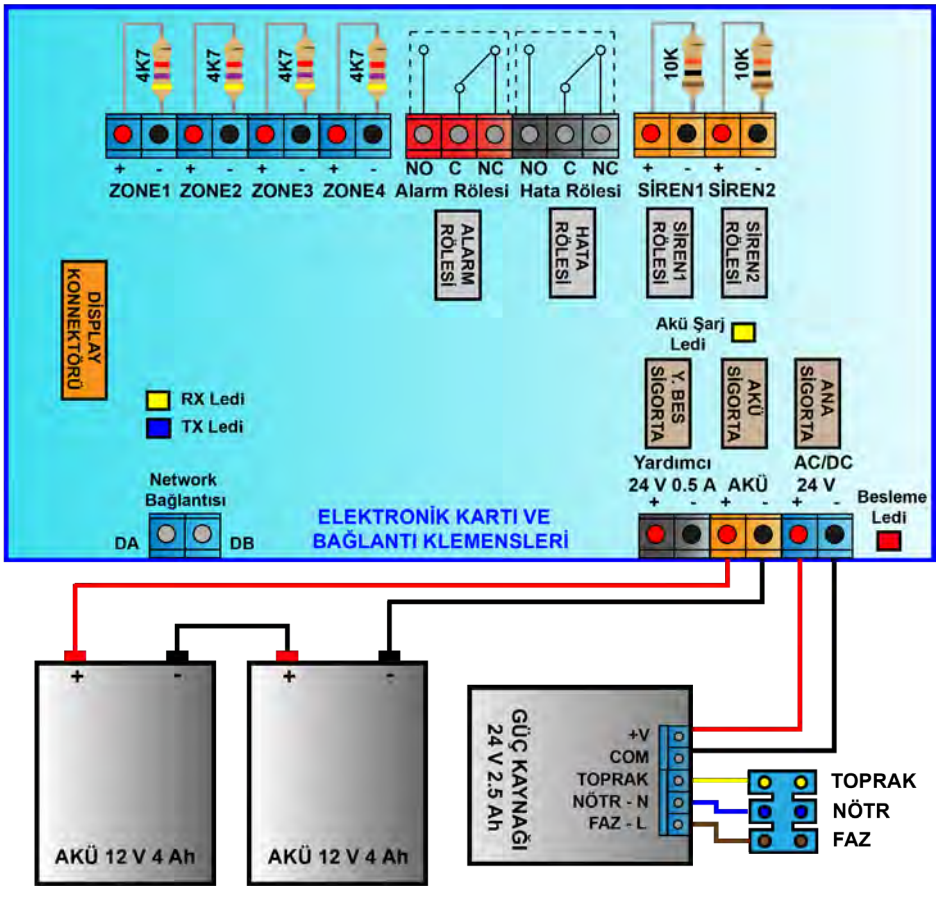
Butonlar ve İşlevleri

- Butonların çalışabilmesi için sistem üzerindeki manuel anahtarın AÇIK konuma getirilmesi gerekmektedir. Kilidin AÇIK konumda olduğu durumda **buzzer** aralıklı olarak çalar.
- **ALARM ON** butonuna basıldığında sirenler ve **buzzer** çalar, yangın LED'i yanar.
- **ALARM OFF** butonuna basıldığında sirenler susar, **buzzer** aralıklı çalar ve alarm iptal LED'i yanar.
- **RESET** butonuna basıldığında sistem resetlenerek normal çalışma moduna döner.
- **TEST/İLERLE** butonu test edilecek bölgeyi belirlemek için kullanılır.
- **AKTİF/PASİF** butonu devre dışı işleminde **zone**un devre dışı veya normal durumunu belirlemek için kullanılır.
- **BUZZER OFF** butonu **buzzer**ı susturmak için kullanılır.

Görsel 4.10'da konvansiyonel sistem yangın alarm santralinin üzerinde genel olarak bulunan giriş soketleri ve işlevleri aşağıda verilmiştir:

- Şebeke voltaj girişi 230 V AC 50 Hz'dir. 5 A'lik sigorta ile korunmaktadır. Toprak bağlantısı mutlaka yapılmalıdır. Topraklama direnci 10 Ω'dan küçük olmalıdır. Santralin besleme kablosu 3x2,5 NYM veya 3x2,5 NYA tipinde olmalıdır.
- Elektrik kesintisinde sistemi beslemek için santral içinde 2 adet 12 V 4 Ah sızdırmaz tip kuru akü kullanılır. Akü girişlerinin doğru bağlanmasına dikkat edilmelidir. Bu giriş 0,5 A cam sigorta (F5 konumlu) ile korunmuştur.
- **Zone** girişleri dedektör ve butonların beslemesinin sağlanmasını, yangın alarmı, hat kısa devre ve hat açık devre sinyallerinin algılanması için kullanılmaktadır. Bir **zone** hattına en fazla 16 adet dedektör ve sınırsız sayıda buton bağlanabilir. 0-500 metre arası bağlantı mesafesi için 1x2x0,8+0,8 JY(st)Y kablo, 500–1500 metre arası için 1x2x1,5+1,5J-Y(st)Y kodlu yangın kablosu kullanılmalıdır.
- Siren 1 ve Siren 2 çıkışı 24 V DC 0,5 A değerindedir. Çıkış 0,5 A'lik cam sigorta (F3 konumlu) ile korunmuştur. Fazla akım çekilmesi çıkışa zarar verebilir. Bu çıkış 10 kΩ 1/4 W hat sonu direnci ile izlenmekte, hat kopuk ve hat kısa devre durumu gösterilmektedir.
- Yardımcı besleme çıkışı 24 V DC 0,5 A'dir. Çıkış 0,5 A'lik cam sigorta (F1 konumlu) ile korunmuştur. Fazla akım çekilmesi santralin hata vermesine, uzun süreli fazla akım çekilmesi ise santralin arızalanmasına sebep olur.
- Yangın alarmı durumu oluştuğunda veya alarm butonuna basıldığında yangın alarm rölesi çıkışı aktif olmaktadır. Aktif olan rölenin eski hâline dönmesi için sistemin reset edilmesi gereklidir. Yangın alarm rölesi sinyal rölesidir, yüksek akımlı anahtarlama yapılması isteniyorsa arada uygun akımlı röle veya kontaktör kullanılmalıdır.
- Sistemde hata durumu oluştuğunda hata rölesi çıkışı aktif olmaktadır. Hata durumu ortadan kalktığında röle eski konumuna dönmektedir. Hata rölesi sinyal rölesidir, yüksek akımlı anahtarlama yapılması isteniyorsa arada uygun akımlı röle veya kontaktör kullanılmalıdır.

Görsel 4.10'da konvansiyonel sistem yangın alarm santralinin üzerinde güç kaynağı ve akü bağlantıları görülmektedir. Buradaki örnek şemada güç kaynağı olarak 24 V 2,5 A **switchmode** kullanılmıştır. Hazır **switchmode** güç kaynağı kullanmak yerine trafo kullanılan modelleri de mevcuttur.



Görsel 4.10: Örnek bir konvansiyonel yangın alarm santralinin besleme gerilimi ve akü bağlantıları

Görsel 4.10'da iki adet 12 V 4 A değerinde birbirine seri bağlı akü kullanılmıştır. Aküler genellikle akım kapasitelerinin %10'u değerinde sarj olur. Şarj gerilim değerleri ise gerilim değerinin 1,2 değeri ile çarpılması sonucunda belirlenir. Örneğin 12 V 4 A akünün şarj gerilimi 12x1.2=14,4 V iken şarj akımı 4x0.1=0.4 A'dir.

Bir akünün bozuk ya da sağlam olduğu aşağıdaki işlem basamaklarına göre kontrol edilir. Örneğin 12 V 4 A'lik akü için;

a) Akü üzerindeki üretim tarihine bakılır. Üretim tarihi 5 yıl öncesine ait ve AVO metre ile ölçülen gerilim değeri 10 V'un altında ise akü bitmiş ya da bozuk kabul edilir.

b) Akü 14,4 V şarj geriliminde ve 0.4 A akım değerinde dolana kadar şarj edilir. "Şarj oldu." ışığı yandığında akü şarjdan çekilir ve üzerindeki gerilim değeri AVO metre ile ölçülür. Ölçülen değer 12,5 V ve altında bir değer ise akü **bitik ya da bozuk** demektir. Fakat ölçülen değer 12,7 V gibi bir değerse "Akü sağlamdır." denir.

c) Akü normal gerilim ve akım değerinde şarj edilir. Ölçülen gerilim değeri 12,7 V olsun. Akü santrale veya başka bir yüke bağlandığında olması gerekenden hızlı boşalıyor ise yine bitik ya da bozuktur.

Görsel 4.10'da kullandığınız akülerin sağlamlığını teyit etmek için akünün sisteme bağlantısını yapıp şarj olmasını bekleyiniz. "Akü dolu" lambası yanınca veya akü bağlı iken bağlantı klemenslerinden akü şarj gerilimini ölçünüz. Ölçtüğünüz değer şarj gerilimine yani (24x1.2=28,8 V) değerine yakın ise akünün santral kartından bağlantısını kesiniz. Aküleri tek tek ölçüp yukarıdaki adımları uygulayınız.

Görsel 4.10'daki devre şemasında 24 V 2,5 A **switchmode** güç kaynağı ve 2 adet 12 V 4 A akü bulunmaktadır. Normal şartlarda 24 V güç kaynağı ile 2 tane seri bağlı 12 V akü şarj edilmez. Çünkü güç kaynağı gerilimi (24 V) ile 2 tane seri bağlı akünün gerilim değeri (12+12=24 V) birbirine eşittir. Güç kaynağı geriliminin ise toplam akü geriliminden 1,2 kat fazla olması yani 24x1,2=28,8 V olması gerekir. Güç kaynağı gerilim değerini yükseltmek için iki yöntem kullanılır.

1. Güç kaynağı olarak kullanılan **switchmode**lar üzerinde, çıkış değerinin %10 oranında artı-eksi yönünde ayarlanmasını sağlayan trimpot ile yapılabilir (Görsel 4.11). Bu yöntem mantıklı değildir. Çünkü güç kaynağı hatasında **switchmode** değişirken değiştiren kişinin bunu bilme şansı çok zayıftır. Ekstra bu sistem için akü tam şarj gerilimi kadar (28.8 V) zaten yükseltilemez. Sağlıklı bir akü şarjı olmaz.

2. Güç kaynağı gerilimini, akü şarj gerilimine artırmak için santral kartı üzerinde step up (DC-DC converter, boost converter devreleri gibi) devreleri kullanılır.



Görsel 4.11: Switch mode güç kaynağı üzerindeki gerilim ayar trimpotu

İlk kurulumda sistem şehir şebekesine bağlanıp çalıştırıldıktan sonra aküler bağlanmalıdır. Birincil güç kaynağı bağlanmadan aküler sistemi çalıştırmayabilir.

4.1.2. Adreslemeli Sistem Yangın Alarm Santrali

Görsel 4.12'de farklı modellerdeki adreslenebilir yangın alarm santralleri görülmektedir.

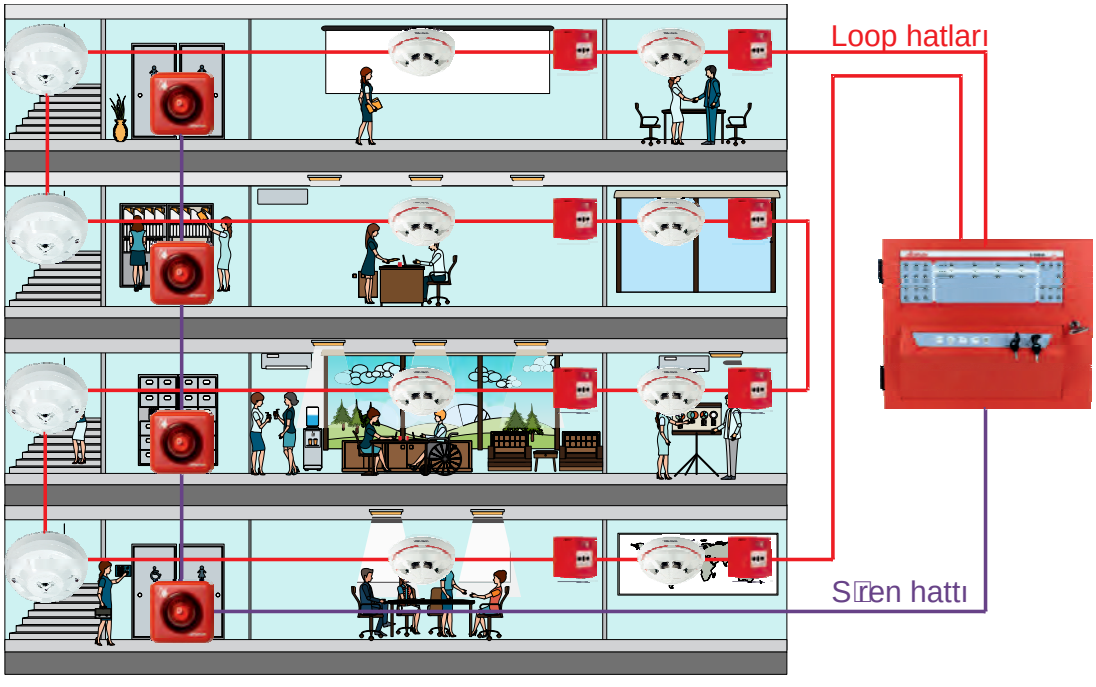


Görsel 4.12: Farklı modellerdeki adreslenebilir sistem yangın alarm santralleri

Adreslenebilir yangın alarm santrali, çıkışında kontrol edeceği **loop** (çevrim) sayısına göre isimlendirilir. Her **loop**'a genel olarak toplam 200 dedektör ve yangın ihbar butonu bağlanabilir. Bu sebeple adreslenebilir yangın alarm santrali, orta ve büyük ölçekli ortamlar için tercih edilir.

Adreslenebilir yangın algılama sistemlerinde, konvansiyonel sistemlerdeki gibi bölge hatları yoktur. Bu sistemlerde dedektörlerin bağlandığı **loop** hatları vardır. Panelden bir kablo çıkar ve sıra ile dedektörleri (adresli duman dedektörü, adresli ısı dedektörü, adresli sıcaklık artış dedektörü, adresli giriş modülü, adresli çıkış modülü vb.) ve yangın ihbar butonlarını dolaşır. Kullanılan toplam algılama elemanı sayısı eğer kullanılan adresli sistemin bir çevriminin kapasitesinden azsa son algılama elemanından sonra kablo tekrar adresli yangın algılama kontrol paneline geri getirilir. Bir çevrimin kapasitesinin üzerinde sayıda algılama elemanı kullanılmışsa ikinci bir çevrim kablosuna geçmek için kablo tekrar panele getirilir. Bir **loop** hattına kapasitesi oranında dedektör ya da yangın ihbar butonu bağlanabilmektedir. Genellikle adresli yangın alarm sistemlerinin bir çevriminin kapasitesi 120 algılama elemanı civarındadır. Son yıllarda artık elektronik teknolojisinin ve kullanılan elektronik malzemelerin gelişimiyle beraber artık bu kapasite oranları 200 ve üzeri sayılara çıkabilmektedir.

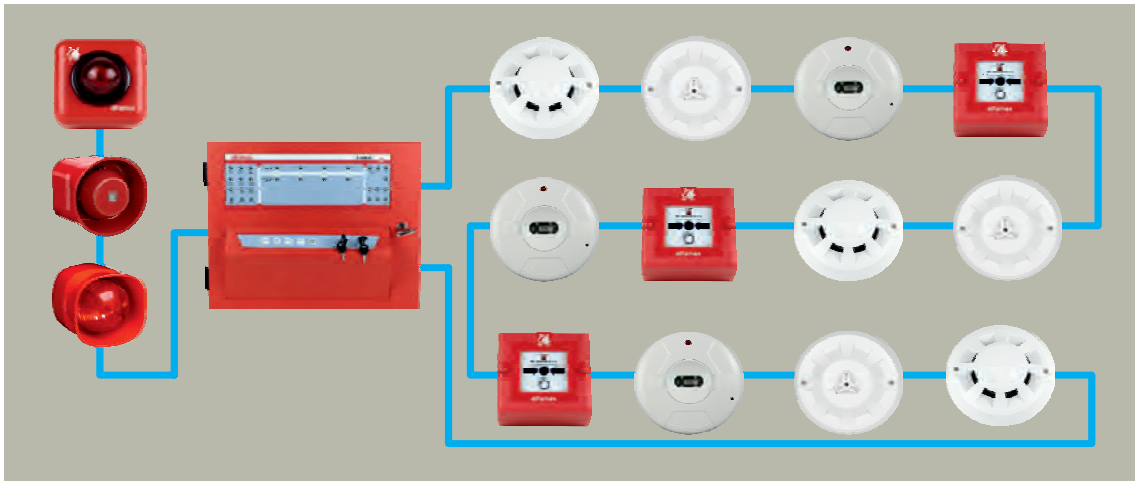
Görsel 4.13'te adreslenebilir bir yangın alarm sisteminin 4 katlı bir binaya yerleşimi ve yangın alarm santrali ile bağlantısı görülmektedir.



Görsel 4.13: 4 katlı bir binada adreslenebilir yangın algılama ve alarm sistemi

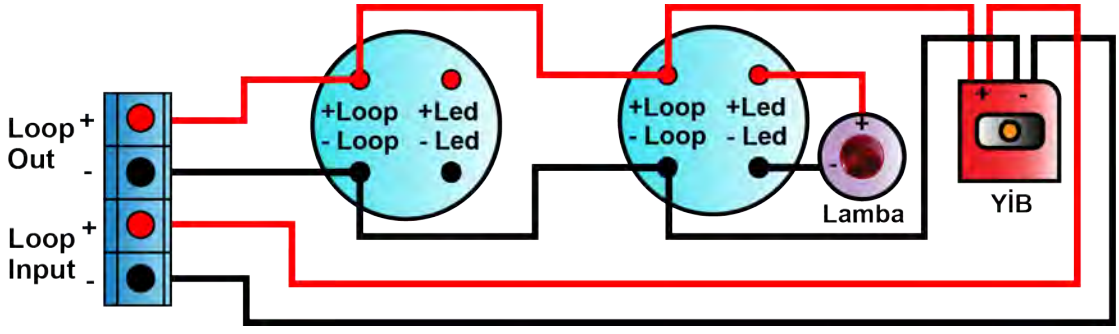
Loopta bulunan dedektör ya da butonlar kendi içinde bölgelere ayrılabilmektedir. Bununla birlikte panel, her bir algılama elemanı ile elektronik olarak haberleşerek anlık durum bilgilerini alabilir. Her algılama elemanına farklı bir adres, numara verilmiştir. Bu sayede her algılama elemanı ile sistem ayrı ayrı haberleşir. Örnek olarak bir yangın ihbar butonuna basıldığında önce ilgili buton alarm sinyalini kendi içinde oluşturmakta, daha sonrasında algılama paneline kendi numarası (adresli) ile beraber göndererek haber vermektedir. Adresli yangın alarm paneli ise bu numarayı kendi hafızasındaki ilgili isim ile eşleştirerek ekranda alarm bilgisini bu yangın ihbar butonunun yerinin ismi ile beraber göstermektedir. Bu durum ise hangi dedektör ya da butondan sinyal geldiğinin tespit edilmesi anlamını taşır.

Görsel 4.14'te bir **loop**lu konvansiyonel yangın alarm sisteminin, yangın alarm santrali ile **loop**lara bağlanacak dedektör ve yangın ihbar butonlarının bağlantısı görülmektedir. **Loop**lara bağlanan dedektörler ve yangın ihbar butonları ile siren çıkışına bağlanan sirenler paralel bağlanmıştır.

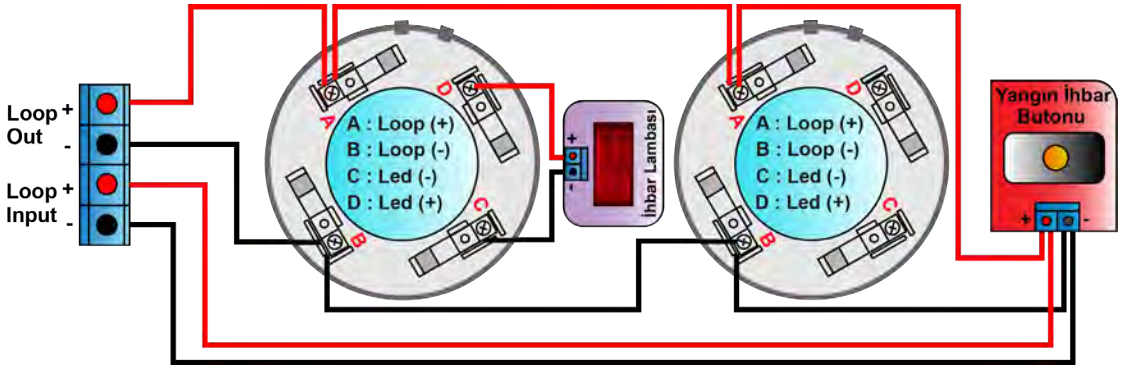


Görsel 4.14: Bir loplu adreslenebilir yangın algılama ve alarm sisteminin bağlantısı

Adreslenebilir yangın alarm santralinin temel bağlantı prensip şeması Görsel 4.15a'da, temel montaj prensip şeması ise Görsel 4.15b'de görülmektedir. **Loop**lara dedektörleri bağlarken yangın alarm santrali ikili yanmaz kablo üzerinden hem enerji hem de data bilgisi gönderir. Kullanılan dedektörlere ve yangın ihbar butonlarına **loop** çıkışından çekilen yanmaz kablo sıra ile girer ve çıkar. En son bağlanan dedektörden sonra **loop** girişine kablo bağlantısı yapılır. Yangın ihbar lambası, dedektör üzerinde boşta kalan uçlara, yönüne dikkat edilerek bağlanır.

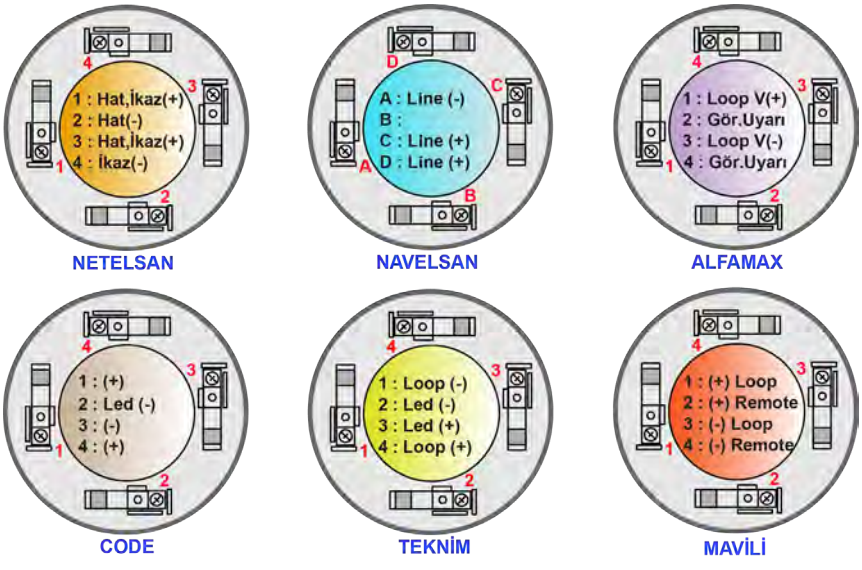


Görsel 4.15a: Adreslenebilir yangın alarm santralinin temel bağlantı şeması



Görsel 4.15b: Adreslenebilir yangın alarm santralinin temel montaj prensip şeması

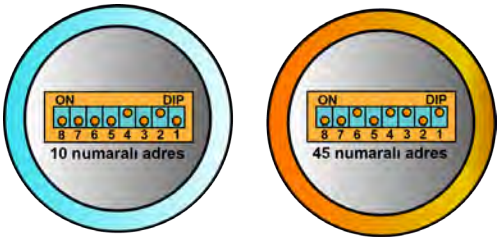
Görsel 4.16'da farklı modellerdeki adreslenebilir dedektörlerin tabanlarındaki bağlantı uçlarının görselleri görülmektedir. İşlevlerine göre bağlantı uçları farklı olsa bile bağlantı mantığı hepsinde aynıdır.



Görsel 4.16: Farklı modellerdeki adreslenebilir dedektörlerin tabanlarındaki bağlantı uçlarının isimleri

Adreslenebilir yangın alarm sistemlerinin en önemli özelliği yangın sinyalinin gönderildiği dedektör veya yangın ihbar butonunun bilinerek yangın noktasına hızlı erişim sağlanmasıdır. Bunun için kullanılan dedektörlere ve yangın ihbar butonlarına adres (numara) bilgisi verilir. Her verilen adrese dayanılarak montaj yapılan mimarinin konum ve yer bilgileri girilir.

Adreslenebilir yangın alarm sistemlerinde önce dedektör ve yangın ihbar butonlarının adresleme işlemi yapılır. Sonra da yangın alarm santralinin bir bilgisayar yazılım programı aracılığı ile kullanılan dedektör ve yangın ihbar butonlarının konum, yer, adres ve hangi **loopa** bağlandığı bilgileri yüklenir. İlk yangın santralleri üretildiğinde adreslenebilir yangın alarm sistemlerinde kullanılan dedektör ve yangın ihbar butonlarının adresleme işlemleri Görsel 4.17'deki gibi **dip switch** anahtarlar ile yapılmaktaydı. Örneğin 10 numaralı dedektör için üzerindeki **dip switch** 00001010 (OFF/OFF/OFF/OFF/ON/OFF/ON/OFF) durumuna, 45 numaralı dedektör için ise **dip switch** 00101101 durumuna alınmalıydı. Konum ve yer bilgileri de yangın alarm santrallerinin üzerine yazılmaktaydı.



Görsel 4.17: Dedektör gövdesinin tabanındaki adresleme dip switch

Son zamanlarda mikrodenetleyici, LCD, grafik ekran ve OLED ekranların, **Wi-Fi** ve bluetooth teknolojilerinin gelişmesiyle adresleme ve konum-yer bilgileri bu cihazlar ile kablolu veya kablosuz haberleşme yoluyla yapılmaktadır. Görsel 4.18a'da **Wi-Fi**li adresleme cihazları, Görsel 4.18b'de ise kablolu adresleme cihazları görülmektedir.





Görsel 4.18a: Wi-Fili adresleme cihazları





Görsel 4.18b: Kablolu adresleme cihazları

Bir **loop**a toplam 255 tane dedektör ve yangın ihbar butonu bağlanabilmektedir. Maksimum 255 sayısının sebebi 8 bit adreslemenin olmasından kaynaklanır. Adresleme işlemi ister kablolu ister kablosuz yapılacak olsun temel mantık aynıdır.

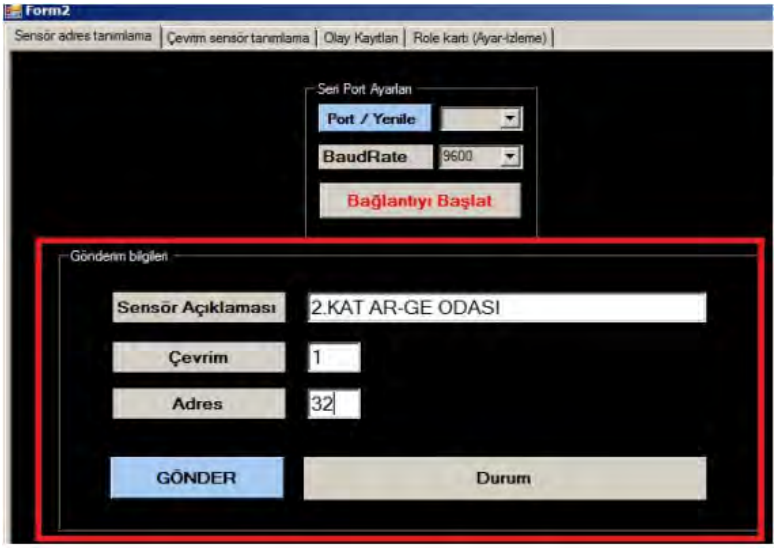
El terminali kablolu ise dedektör için **Loop+** ve **Loop**- uçlarına, yangın ihbar butonları için **artı** ve **eksi** uçlarına cihaz bağlanır. Adresleme el terminali çalıştırılır. Dedektör veya yangın ihbar butonunun adres bilgisi "OKU-READ" komutu ile okunur. Adres bilgisi doğru değil ise veya hiç atanmamış ise adres numarası (0-255 arasında bir sayı) girilir ve "WRITE-OK-ONAY" tuşuna basılır. "Adresleme işlemi tamamlandı." mesajı alındıktan sonra diğer cihazlar da aynı şekilde numaralandırılabilir.

El terminali kablosuz ise dedektör veya yangın ihbar butonu yan yana getirilir. İki cihaz arasında bağlantı kurulduktan sonra adres bilgisi var ise okunabilir veya hiç atanmamış ise adres numarası girilerek "WRITE-OK-ONAY" tuşuna basılarak adresleme işlemi tamamlanır. Diğer cihazlarda da aynı şekilde adres ataması yapılabilir.

Yangın alarm santrali üzerinden adresleme işlemi yapılırken dedektör veya yangın ihbar butonunun yangın alarm santraline ekstra verilen bağlantı kablosu ile bağlantısı yapılır. Menü seçeneklerinden adresleme sekmesine gelinir. Adres bilgisi var ise okunabilir veya hiç atanmamış ise adres numarası girilerek "WRITE-OK-ONAY" tuşuna basılır ve adresleme işlemi tamamlanır.

Teknolojinin gelişmesiyle birlikte yangın alarm santralleri üreticileri, LCD ve grafik ekranlarda metin ifadelerini gösteren ve bilgisayar ile haberleşen, bir arayüz programı ile adreslenebilen santraller geliştirmiştir. Görsel 4.19'da örnek bir bilgisayar arayüz programının ekran görüntüsü görülmektedir.

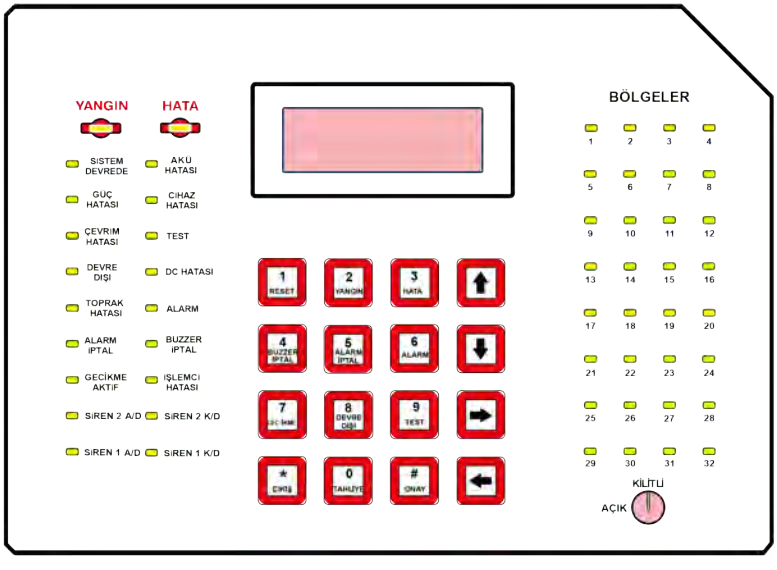
Yangın alarm santralleri bir bilgisayar programı ile programlanırken önce santralin PC ile kablo bağlantısı (USB veya seri port üzerinden) yapılır. Bilgisayar arayüz programı çalıştırılır. Bilgisayar ile yangın alarm santralinin bağlantısı, arayüz programında port ve baund rate değeri girilip "Bağlantıyı başlat." sekmesine tıklanarak sağlanır. Daha sonra arayüz programı ile ortamın montajında kullanılan dedektör ve yangın ihbar butonlarının adresleri ile konum, yer, adres ve **loop** bilgileri bütün hepsi için sıra ile girilir. Arayüz programının özelliğine göre "tek tek" veya "Toplu olarak gönder." ikonuna tıklanarak girilen bilgiler yangın alarm santraline yüklenir.



Görsel 4.19: Örnek bir adresleme bilgisayar arayüz programının ekran görüntüsü

Yangın alarm santralinin ve kullanılacak dedektör ve yangın ihbar butonlarının adreslenmesi ve programlanması tamamlandıktan sonra bilgisayar ile bağlantısı kesilir. RESET tuşuna basılarak yangın alarm santrali resetlenir. Santral resetlendikten sonra yeniden başlar. Montajı yapılan sistemin denemeleri yapıldıktan sonra sistem devreye alınarak teslim edilir.

Görsel 4.20'de adreslenebilir sistem yangın alarm santralinin örnek ön yüzü görülmektedir.



Görsel 4.20: Adreslenebilir sistem yangın alarm santralinin örnek ön yüzü

Adreslenebilir sistem yangın alarm santralinin üzerindeki göstergeler genel olarak aşağıda verilmiştir:

. **Sistem devrede LED'i**, santrale enerji geldiği sürece yanar.

. **Hata LED'i**, santraldeki ve cihazlardaki hata durumlarında yanar, hatanın ortadan kalkmasıyla söner.

. **Yangın LED'i**, cihazlardan yangın alarm sinyali geldiğinde veya el ile yangın alarmı verildiğinde yanar.

. **Cihaz hatası LED'i**, herhangi bir çevrime bağlı cihazın kısa devre olması, açık devre olması veya yerinden çıkarılması durumunda yanar. Hatanın ortadan kalkmasıyla söner.

. **Akü hatası LED'i**, akü bağlantısı olmadığı veya akü voltajı 18 V altında ise yanar. Akü bağlantısı yapıldığında akü yeterli gerilime sahipse (>18 V) akü hata LED'i söner.

. **Güç hatası LED'i,** santralde şebeke gerilimi bulunmadığı zaman yanar. Şebeke gerilimi geldiğinde bu LED söner.

. **Çevrim hatası LED'i**, santral 5 dk. süre ile çevrim kartıyla haberleşemediğinde çevrim kısa devre veya açık devre olduğunda yanar. Bu hata durumları ortadan kalktığında söner.

. **Test LED'i**, herhangi bir cihaz, çevrim veya bölge test durumuna alındığında yanar.

. **Devre dışı LED'i**, cihaz; çevrim, bölge ve çıkışlardan herhangi biri devre dışı bırakıldığı zaman yanar ve iptal edilene kadar yanmaya devam eder.

. **DC hatası LED'i**, 24 V DC çıkışından fazla akım çekildiğinde yanar.

. **Toprak hatası LED'i**, toprak hattında voltaj kaçağı olması durumunda yanar.

. **Alarm LED'i**, çevrimde bulunan herhangi bir cihazdan alarm gelmesi sonucu yanar ve sirenler çalar.

. **Alarm iptal LED'i,** sirenlerin susturulması durumunda yanar.

. **Buzzer iptal LED'i**, santral üzerinde bulunan **buzzer**ın susturulması sonucu yanar.

. **Gecikme aktif LED'i**, cihaz, çevrim veya bölgeye gecikme verilmesi durumunda yanar.

. **İşlemci hatası LED'i**, ana kart üzerinde bulunan işlemcinin arızalanması sonucu yanar.

. **Siren 1 K/D LED'i,** siren 1'de bulunan sirenlerden herhangi birinin kısa devre olması sonucu yanar ve hata giderildiğinde söner.

. **Siren 1 A/D LED'i**, siren 1'de bulunan sirenlerden herhangi birinin açık devre olması durumunda yanar ve hata giderildiğinde söner.

. **Siren 2 K/D LED'i**, siren 2'de bulunan sirenlerden herhangi birinin kısa devre olması sonucu yanar ve hata giderildiğinde söner.

. **Siren 2 A/D LED'i**, siren 2'de bulunan sirenlerden herhangi birinin açık devre olması durumunda yanar ve hata giderildiğinde söner.

Adreslenebilir sistem yangın alarm santralinin üzerinde genel olarak bulunan butonlar ve işlevleri aşağıda verilmiştir:

. Anahtar KAPALI durumda iken sadece yön tuşları ve çıkış tuşu çalışır. Anahtar AÇIK durumda olduğunda tuş takımı kullanılırken karşılık gelen özellikler devre dışı kalır.

. **RESET** butonuna basıldığında santral resetlenir. Tuş takımında 1 numaradır.

. **YANGIN** butonuna basıldığında ekranda yangın alarm sayısı ve yangın alarm uyarısı veren cihazlar görülebilir. Tuş takımında 2 numaradır.

. **HATA** butonuna basıldığında ekranda hata sayısı ve hata bilgileri görülebilir. Tuş takımında 3 numaradır.

. **BUZZER İPTAL** butonu **buzzerı** susturmak için kullanılır. Anahtar AÇIK durumunda çalan kesikli **buzzer** sesini susturmaz. Tuş takımında 4 numaradır.

. **ALARM İPTAL** butonuna basıldığında sirenler susar ve alarm iptal LED'i yanar. Tuş takımında 5 numaradır.

. **ALARM** butonuna basıldığında sirenler ve **buzzer** çalar, yangın LED'i yanar. Tuş takımında 6 numaradır.

. **GECİKME** butonuna basıldığında ekranda gecikme bilgisi görülebilir. Tuş takımında 7 numaradır.

. **DEVRE DIŞI** butonuna basıldığında ekranda devre dışı bilgisi görülebilir. Tuş takımında 8 numaradır.

. **TEST** butonuna basıldığında santral üzerindeki tüm LED'ler yanar ve 5 sn. sonra söner. Tuş takımında 9 numaradır.

. **ÇIKIŞ** butonuna basıldığında ekran ilk durumuna döner.

. **TAHLİYE** butonu programlanabilir butondur. Tuş takımında 0 numaradır.

. **ONAY** butonu seçimlerin onayı için kullanılır.

. **İLERİ** butonuna basıldığı zaman ekranda menüye giriş yapılır.

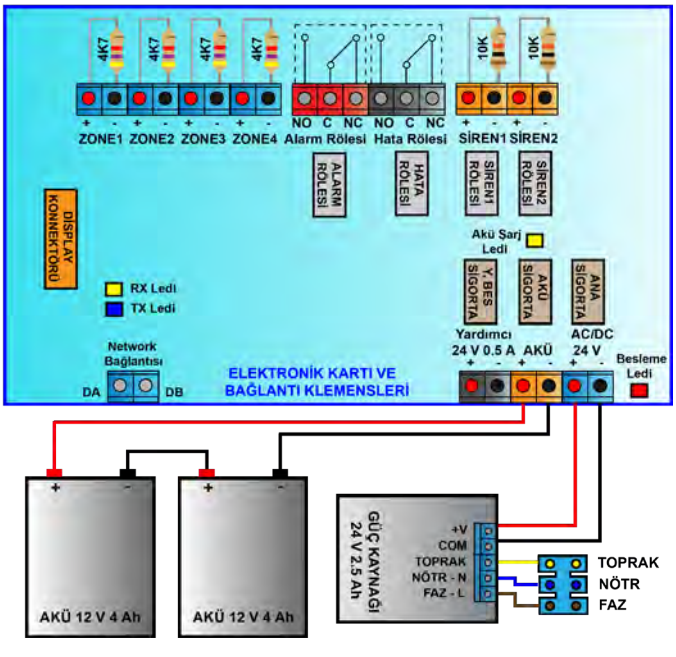
. **YUKARI** butonuna basıldığı zaman menü seçim çubuğu yukarı gider.

. **AŞAĞI** butonuna basıldığı zaman menü seçim çubuğu aşağı gider.

. **GERİ** butonuna basıldığı zaman ekran bir önceki menüye döner.

Görsel 4.21'de örnek bir adreslenebilir yangın alarm santralinin besleme gerilimi ve akü bağlantıları görülmektedir. Buradaki örnek şemada güç kaynağı olarak 24 V 4,5 A **switchmode** kullanılmıştır. Hazır **switchmode** güç kaynağı kullanmak yerine trafo kullanılan modelleri de mevcuttur.

Yine şemada 2 adet 12 V 7 A değerindeki akü seri bağlanarak kullanılmıştır. Aküler genellikle akım kapasitelerinin %10'u değerinde sarj olur. Şarj gerilim değerleri ise gerilim değerinin 1,2 ile çarpılması ile belirlenir. Örneğin 12 V 7 A akünün şarj gerilimi 12x1,2=14,4 V iken şarj akımı 7x0,1=0,7 A'dir.



Görsel 4.21: Örnek bir adreslenebilir yangın alarm santralinin besleme gerilimi ve akü bağlantıları

Adreslenebilir sistem yangın alarm santralinin üzerinde genel olarak bulunan giriş soketleri ve işlevleri aşağıda verilmiştir:

- Şebeke voltaj girişi 230 V AC 50 Hz'dir. Toprak bağlantısı mutlaka yapılmalıdır.
- Elektrik kesintisinde santrali beslemek için santral içinde 2 adet 12 V 7 Ah sızdırmaz tip, kurşun asitli akü kullanılır. Akü girişlerinin doğru bağlanmasına dikkat edilmelidir. Bu giriş 4 A cam sigorta ile korunmaktadır.
- Çevrim girişleri dedektör, buton, siren ve izolatörün beslemesinin sağlanması, yangın alarmı, hat kısa devre ve hat açık devre sinyallerinin algılanması için kullanılmaktadır. Bir çevrim hattına en fazla 128 adet cihaz bağlanabilir. 500 ile 1500 metre arası bağlantı mesafesinde 1x2x1,5J-Y(st)Y kodlu yangın kablosu kullanılmalıdır.

Adreslenebilir sistem yangın alarm santralinin üzerinde genel olarak bulunan çıkış soketleri ve işlevleri aşağıda verilmiştir:

- Siren 1 çıkışı 24 V DC 0,5 A değerindedir. Çıkış 0,5 A'lik cam sigorta ile korunmaktadır. Fazla akım çekilmesi çıkışa zarar verebilir ve bu çıkış 47 kΩ 1/4 W hat sonu direnci ile izlenmekte, hat kopuk ve hat kısa devre durumu gösterilmektedir.
- Siren 2 çıkışı 24 V DC 0,5 A değerindedir. Çıkış 0,5 A'lik cam sigorta ile korunmaktadır. Fazla akım çekilmesi çıkışa zarar verebilir ve bu çıkış 47 kΩ 1/4 W hat sonu direnci ile izlenmekte, hat kopuk ve hat kısa devre durumu gösterilmektedir.
- Yardımcı besleme çıkışı 24 V DC 0,5 A'dir. Çıkış 0,5 A'lik cam sigorta ile korunmaktadır. Fazla akım çekilmesi santralin hata vermesine neden olabilir. Uzun süreli fazla akım çekilmesi santralin arızalanmasına sebep olur.
- Yangın alarmı durumu oluştuğunda veya alarm butonuna basıldığında yangın alarm rölesi çıkışı aktif olmaktadır. Aktif olan rölenin tekrar eski konuma dönmesi için santralin resetlenmesi gerekmektedir. Yangın alarm rölesi sinyal rölesidir, yüksek akımlı anahtarlama yapılması istenildiği takdirde arada uygun akımlı röle veya kontaktör kullanılmalıdır.
- Sistemde hata durumu oluştuğunda hata rölesi çıkışı aktif olmaktadır. Hata durumu ortadan kalktığında röle eski konumuna tekrar dönmektedir. Hata rölesi sinyal rölesidir ve yüksek akımlı anahtarlama yapılması istenildiği takdirde araya uygun akımlı röle veya kontaktör kullanılmalıdır.

4.2. YANGIN ALARM SİSTEMİ DONANIMLARI

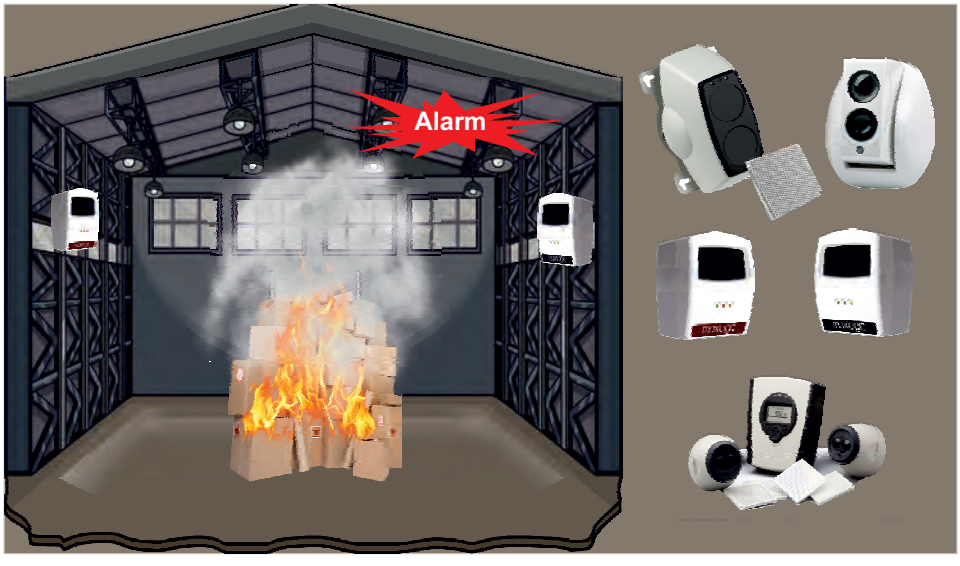
Yangın alarm sistemlerinde kullanılan donanımlar seçilen yangın alarm santraline göre farklılıklar göstermektedir. Tercih edilen yangın alarm santrali ister konvansiyonel sistem ister adresli sistem olsun kullanılacak ürünlerin temel çalışma işlevleri aynı olsa da ürünler arasında bağlantı ve kurulum farklılıkları vardır.

4.2.1. Duman Dedektörü

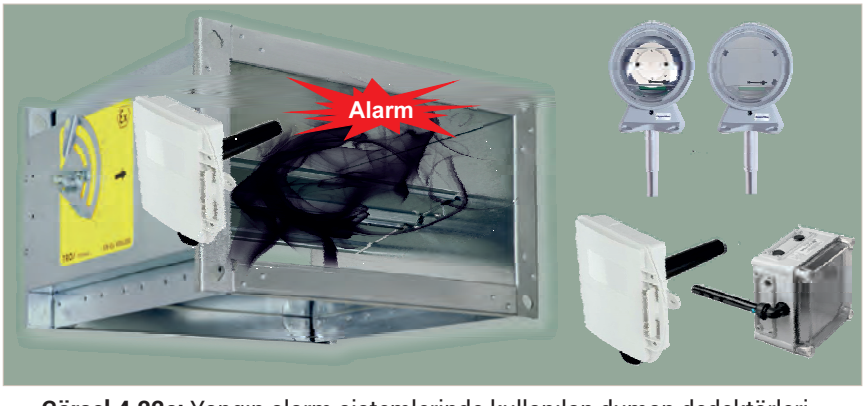
Yanma olayının sonucunda ortaya çıkan duman unsurunu algılayan dedektörlerdir. Duman dedektörü, dumanlı yangınlarda duman zerreciklerini algılama yöntemi ile çalışmaktadır. Görsel 4.22a, 4.22b ve 4.22c'de yangın alarm sistemlerinde kullanılan farklı modellerdeki duman dedektörlerinin görselleri görülmektedir.



Görsel 4.22a: Yangın alarm sistemlerinde kullanılan duman dedektörleri



Görsel 4.22b: Yangın alarm sistemlerinde kullanılan duman dedektörleri



Görsel 4.22c: Yangın alarm sistemlerinde kullanılan duman dedektörleri

Duman dedektörlerinin çeşitleri ve kullanım yerleri aşağıdaki gibi sınıflandırılır:

a) Optik duman dedektörü; konferans odaları, bilgi işlem odaları, büyük mağazalar, fabrikalar elektrik/mekanik odaları, yatak odaları, kütüphaneler, kayıt stüdyoları, okullar, tiyatro sahneleri, restoranlar, yemekhanelerde kullanılabilir.

b) Işın tipi duman dedektörü (beam dedektör); yüksek hacimli depolar ve imalat holleri, sinema, tiyatro salonları, koridorlarda kullanılabilir.

c) Aktif hava emmeli duman dedektörü; telekomünikasyon odaları, bilgi işlem odaları, tarihî eser ler, müzeler, soğuk hava depoları, asansör şaftlarında kullanılabilir.

Duman dedektörlerinin montajı esnasında dikkat edilmesi gerekenler

- Dedektörler toza, neme, darbeye, düşmeye ve suya karşı korunmalıdır.
- Dedektör tabanında bulunan montaj delikleri yardımıyla taban duvara sabitlenir. Tesisat kabloları ilgili yerlere gevşek olmayacak şekilde bağlanmalıdır.

4.2.2. Isı (Sıcaklık) Dedektörü

Yanma olayının sonucunda ortaya çıkan sıcaklık unsurunu algılayan dedektörlerdir. Sıcaklık dedektörleri, ortamdaki sıcaklığın belirli seviyeyi aşması hâlinde hızlı sıcaklık artışını algılayarak alarm verir. Binalar, evler, oteller, restoranlar, bürolar, okullar, depolar, kütüphaneler vb. birçok yerde kullanılabilir. Görsel 4.23'te yangın alarm sistemlerinde kullanılan farklı modellerdeki ısı dedektörlerinin görselleri görülmektedir.



Görsel 4.23: Yangın alarm sistemlerinde kullanılan farklı modellerdeki ısı dedektörleri

Isı dedektörleri; sabit sıcaklık dedektörü, sıcaklık artış dedektörü ve sabit sıcaklık-sıcaklık artış dedektörleri olarak sınıflandırılır.

Isı dedektörlerinin montajı esnasında dikkat edilmesi gerekenler

a) Dedektörler toza, neme, darbeye, düşmeye ve suya karşı korunmalıdır.

b) Dedektör tabanında bulunan montaj delikleri yardımıyla taban duvara sabitlenir. Tesisat kabloları ilgili yerlere gevşek olmayacak şekilde bağlanmalıdır.

c) Dedektör gövdesi taban üzerine sadece bir şekilde yerleştirilebilmektedir. Dedektör gövdesi tabana yerleştirilip saat yönünde çevrilerek montaj yapılır.

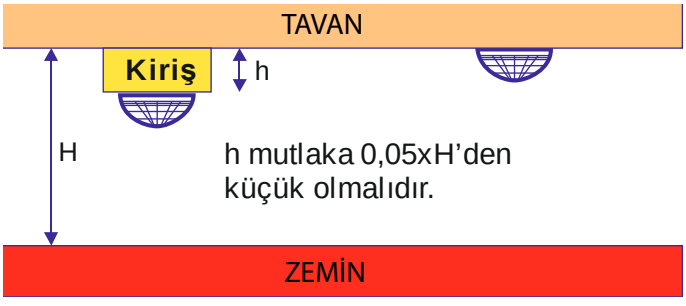
ç) Sıcaklık dedektörü 5 metre çapında bir alanda algılama yapabilir.

d) Koridor genişliği 2 metreden küçük koridorlarda 10 metre ara ile 2 metreden büyük koridor larda ise 7 metre ara ile monte edilmelidir.

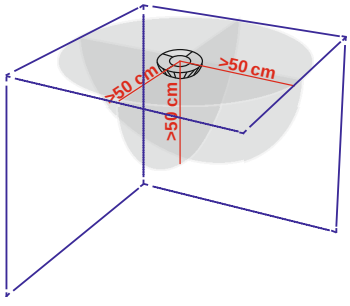
e) Dedektör içindeki algılama elemanının sağlıklı çalışması için dedektör tavana gömülmemeli ve oda yüksekliğinin üst %5'i içerisinde kalacak şekilde monte edilmelidir (Görsel 4.24a).

f) Dedektör hava girişinden veya havalandırma menfezlerinden 1 metre uzağa takılmalıdır.

g) Dedektörlerin alt kısımlarında bütün yönlere doğru en az 50 cm'lik açıklık olmalıdır (Görsel 4.24b).



Görsel 4.24a: Isı dedektörünün tavan-yer arasına montajı



Görsel 4.24b: Isı dedektörünün tavan-kenarlar arasına montajı

4.2.3. Multi (Kombine) Dedektör

Yanma olayının sonucunda ortaya çıkan hem duman hem de sıcaklık unsurunu algılayan dedektörlerdir. Multi dedektör, dumanlı yangınlarda duman zerreciklerini algılamaktadır. Aynı zamanda ortamdaki sıcaklığın belirli seviyeyi aşması durumunda hızlı sıcaklık artışını algılayarak alarm vermektedir. Multi dedektör, kombine veya duman+ısı dedektörü olarak da bilinir. Bina, ev, otel, restoran, büro, okul, depo, kütüphane vb. birçok yerde kullanılabilir. Görsel 4.25'te yangın alarm sistemlerinde kullanılan farklı modellerdeki multi dedektörlerin görselleri görülmektedir.



Görsel 4.25: Yangın alarm sistemlerinde kullanılan farklı modellerdeki multi dedektörler

Multi dedektörler hem duman hem ısı unsurunu algıladıkları için montajı esnasında dikkat edilmesi gerekenler, ısı dedektörü ile aynıdır.

4.2.4. Alev Dedektörü

Yanma olayının sonucunda ortaya çıkan alev unsurunu algılayan dedektörlerdir. Yanma esnasında alevin çıkardığı morötesi (UV) ışınları algılar. Cisimlerin üzerlerinde meydana gelebilecek olan ışık parlamalarını veya kızılötesi ışık kaynaklarını algılamaz. Dedektör 6 metre mesafeden 3 cm uzunluğundaki alevi algılayıp alarm üretebilir. Yüksek tavanlı yerlerde, yangın riskinin daha çok olduğu ortamlarda, patlayıcı ve kolay yanıcı maddelerin bulunduğu ortamlarda kullanılır. Görsel 4.26'da yangın alarm sistemlerinde kullanılan farklı modellerdeki alev dedektörlerinin görselleri görülmektedir.



Görsel 4.26: Yangın alarm sistemlerinde kullanılan farklı modellerdeki alev dedektörü

4.2.5. Dedektör Taban Ayağı ve Bağlantıları

Yangın alarm sistemlerinde kullanılan dedektörlerden duman, ısı, multi (duman+ısı) ve alev dedektörlerinin seçilen modele göre taban ayağı ile gövdesi aynı büyüklüktedir. Tercih edilen yangın alarm santrallerine bağlantıları da kendi içinde aynı olduğu için birbirleri yerine takılabilir.

Görsel 4.27a'da farklı modelde dedektör taban ayakları, Görsel 4.27b'de ise ışıklı, sirenli ve ışıklı-sirenli taban ayaklı dedektör görselleri görülmektedir. Son zamanlarda dedektör taban ayakları ışıklı, sirenli ve ışıklı-sirenli olarak üretilmektedir.



a) Dedektör taban ayakları

b) Derin taban ayakları

c) Asma tavan ünitesi

d) Iş ıklı, sirenli ve ış ıklı-sirenli taban ayaklı dedektör resimleri

e) Sirenli taban ayak

Görsel 4.27a: Farklı modelde dedektör taban ayakları

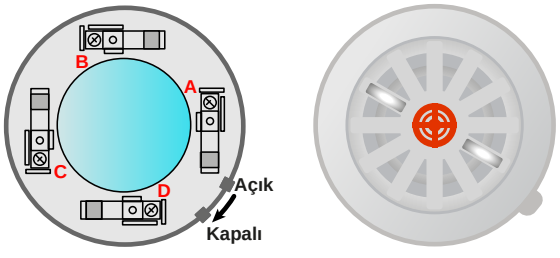


Görsel 4.27b: Işıklı, sirenli ve ışıklı-sirenli taban ayaklı dedektörler

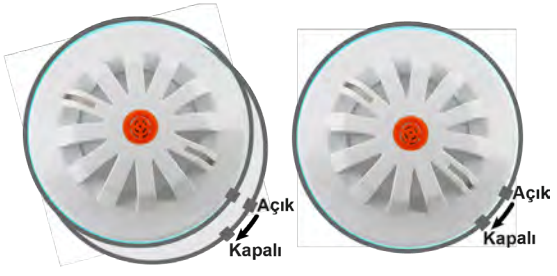
Yangın alarm sisteminde kullanılan dedektörler genel olarak taban ayağı ve gövde olmak üzere iki kısımdan oluşur. Taban ayaklarının tavana montajı yapılır. Dedektör bağlantıları, taban ayağındaki bağlantı noktalarına yapılır. Önce taban ayağı montajı, kablo bağlantısı ve üzerine gövdenin oturtulması ile dedektörün montajı tamamlanır.

Dedektör gövdesinin tabana sabitlenmesi iki aşamada olur. Birinci aşamada dedektörün gövdesi ve taban ayağı yan yana konur. İkinci aşamada dedektörün gövdesi taban ayağının üzerine, bastırılmadan taban ayağındaki işarete dikkat edilerek konur. Gövde "**Kapalı**" yazısına doğru sağa çevrilerek gövde üzerindeki çizgi ile taban ayağındaki çizginin üst üste gelmesi sağlanır. Böylelikle dedektörün gövdesi taban ayağına kilitlenerek sabitleme yapılmış olur.

Görsel 4.28a'da dedektör tabanının ve gövdesinin üstten görünüşü, Görsel 4.28b'de ise gövdenin tabana sabitlenme aşamaları görülmektedir.

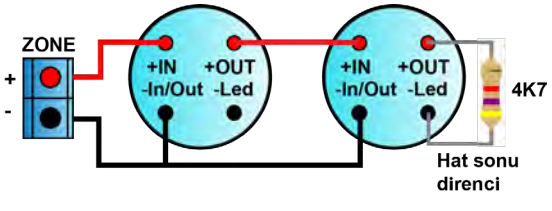


Görsel 4.28a: Dedektör tabanının ve gövdesinin üstten görünüşü

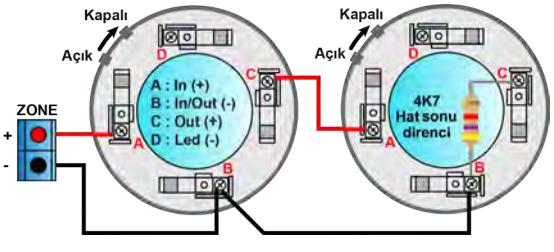


Görsel 4.28b: Gövdenin tabana sabitlenme aşamaları

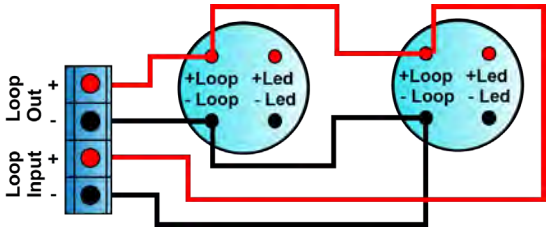
Yangın alarm sistemlerinde kullanılan dedektörlerden duman, ısı, multi (kombine, duman+ısı) ve alev dedektörlerinin bağlantıları seçilen santral türüne göre aynıdır. Modele göre taban ayağının bağlantı uçlarının yeri değişebilir. Görsel 4.29a'da konvansiyonel dedektörlerin bağlantı şeması, Görsel 4.29b'de montaj şeması görülmektedir. Görsel 4.30a'da adreslenebilir dedektörlerin bağlantı şeması, Görsel 4.30b'de montaj şeması görülmektedir.



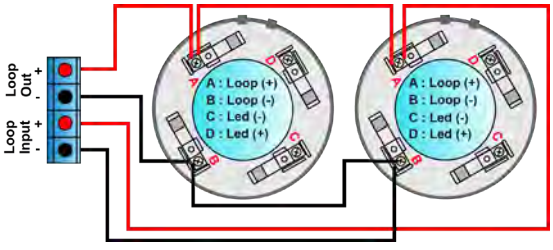
Görsel 4.29a: Konvansiyonel dedektörlerin montaj şeması



Görsel 4.29b: Konvansiyonel dedektörlerin bağlantı şeması



Görsel 4.30a: Adreslenebilir dedektörlerin bağlantı şeması



Görsel 4.30b: Adreslenebilir dedektörlerin montaj şeması

4.2.6. Yangın İhbar Butonu

Yangın ihbar butonu, yangın alarm santraline el ile yangın var bilgisini göndermek için kullanılır. Yangın ihbar butonu sadece iç kısımlarda kullanılmaya uygundur. Sıva altı veya sıva üstü montajlanabilir. İlk durumuna döndürülebilme (resetlenebilme) özelliğine sahiptir.

Görsel 4.31'de yangın alarm sistemlerinde kullanılan farklı modellerdeki yangın ihbar butonlarının görselleri görülmektedir.



Görsel 4.31: Yangın alarm sistemlerinde kullanılan farklı modellerdeki yangın ihbar butonları

Yangın ihbar butonlarının montajı esnasında dikkat edilmesi gerekenler

a) Kullanım sırasında santral toza, neme, darbeye, düşmeye ve suya karşı korunmalıdır.

b) Yangın ihbar butonları maksimum 30 m aralıklarla montaj yapılmalıdır.

c) Yangın ihbar butonları duvara yer seviyesinden 1,2 m ile 1,6 m arası yüksekliğe monte edilmelidir.

ç) Yangın ihbar butonları kaçış yolları üzerinde açılan kapıların içine ve dışına monte edilmelidir.

Görsel 4.32a'da konvansiyonel, Görsel 4.32b'de ise adreslenebilir yangın ihbar butonu bağlantısı görülmektedir. Konvansiyonel sistemde, yangın ihbar butonu arkasında bulunan ikili klemense **zone** kabloları takılarak bağlantı gerçekleştirilir. Bağlantı yönsüzdür bu nedenle istenildiği gibi bağlanabilir. Bir **zone**a istenilen sayıda yangın ihbar butonu bağlanabilir.

Adresli sistemde yangın ihbar butonu arkasında bulunan ikili klemense çevrim kabloları takılarak bağlantı gerçekleştirilir. Çevrim kablosunun (+) ucu butonun (+) ucuna, (-) ucu da butonun (-) ucuna bağlanmalıdır.

4.2.7. Siren

Yangın alarm santrali, dedektörlerden veya yangın ihbar butonlarından "Yangın var." bilgisi geldiğinde sirenleri aktif hâle getirir. Siren ise yangın alarm santralinden gelen tetikleme sinyaline göre sesli ve ışıklı yangın alarm bilgisi vererek insanları uyarır. Sirenlerin flaşörlü (ışıklı) ve flaşörsüz (ışıksız) çeşitleri vardır. Ayrıca iç mekânlarda kullanılmak üzere hazırlanan sirenlere **dâhilî**

siren, dış mekânlarda kullanılmak üzere hazırlanan sirenlere ise **haricî siren** denir. Görsel 4.33'te yangın alarm sistemlerinde kullanılan farklı modellerdeki siren görselleri görülmektedir.



Görsel 4.33: Yangın alarm sistemlerinde kullanılan farklı modellerdeki sirenler

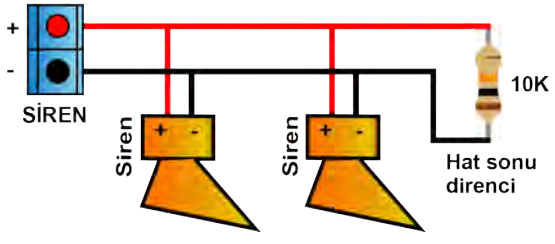
Sirenlerin montajı esnasında dikkat edilmesi gerekenler

a) Kullanım sırasında siren toza, neme, darbeye, düşmeye ve suya karşı korunmalıdır.

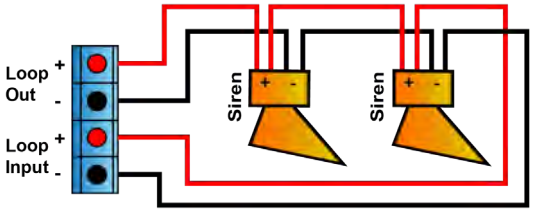
b) Flaşörlü veya flaşörsüz siren sadece iç kısımlarda kullanılmaya uygundur. Sıva altı veya sıva üstü montajlanabilir, yönlü bağlantıya sahip olduğundan doğru bağlantı yapılmalıdır.

Görsel 4.34a'da konvansiyonel sistemde kullanılan siren bağlantısı görülmektedir. Bu bağlantı şeması aynı zamanda adreslenebilir yangın alarm santrallerinde ayrı bir siren çıkışı olan santrallerde de kullanılır. Görsel 4.34b'de adreslenebilir sistemde **loop** hattına siren bağlantısı görülmektedir. Sirenin **loop** hattına bağlantısı 10 adedi geçmemelidir.

Konvansiyonel sistemde, yangın alarm sireninin arkasında bulunan ikili klemense, yangın alarm santralinin siren çıkışları (+) ve (-) uçlar karşılıklı gelecek şekilde bağlantı yapılır. Adresli sistemlerde ise yangın alarm sireninin arkasında bulunan ikili klemense, çevrim kablosunun uçları (+) uç (+) uca (-) uç (-) uca gelecek şekilde karşılıklı bağlanmalıdır.



Görsel 4.34a: Konvansiyonel sistem siren bağlantısı

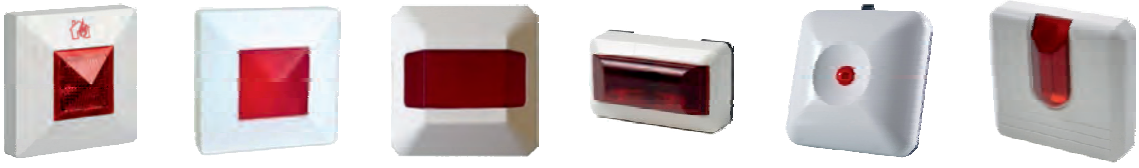


Görsel 4.34b: Adreslenebilir sistem siren bağlantısı

4.2.8. Paralel İhbar Lambası

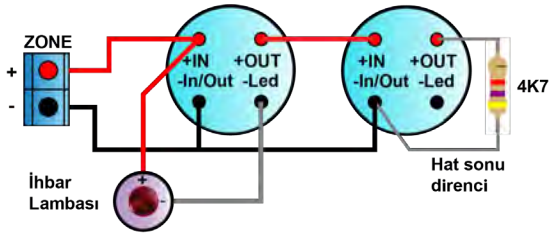
Yangın algılama dedektörlerinin; yüksek tavan, asma tavan, yükseltilmiş taban, girilmesi sakıncalı odalar vb. yerlere montajından dolayı alarm LED'lerinin görülemediği durumlarda, ek bildirim LED'i olarak kullanılır. Yani paralel ihbar lambası ile alarm veren dedektörün fark edilmesini kolaylaştırmak amaçlanır. Son zamanlarda paralel ihbar lambalarının işlevleri artırılarak sesli ve sesli-ışıklı olmak üzere üretilmektedir.

Görsel 4.35'te yangın alarm sistemlerinde kullanılan farklı modellerdeki paralel ihbar lambasının görselleri görülmektedir.

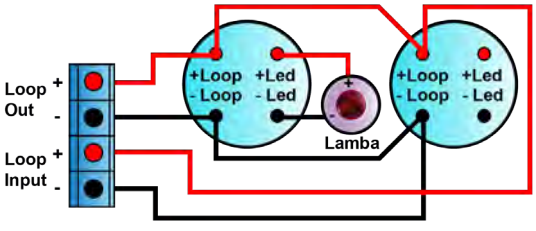


Görsel 4.35: Yangın alarm sistemlerinde kullanılan farklı modellerdeki paralel ihbar lambaları

Görsel 4.36a'da konvansiyonel ve Görsel 4.36b'de adreslenebilir sistemlere paralel ihbar lambalarının bağlantısı görülmektedir. Görsel 4.36a'da paralel ihbar lambası +IN ile –LED uçları arasına, Görsel 4.36b'de ise +LED ile –LED arasına bağlanmıştır.



Görsel4.36a: Konvansiyonel sistem paralel ihbar lambası bağlantısı



Görsel 4.36b: Adreslenebilir sistem ihbar lambası bağlantısı

4.2.9. Doğal Gaz– LPG (Likit Petrol Gazı) Dedektörü

Doğal gaz veya LPG kullanılan yerlerde herhangi bir sebepten dolayı gaz kaçağı olması ve ortamdaki gaz seviyesinin belirlenen seviyeyi aşması durumunda sistem alarm vermektedir. Görsel 4.37'de yangın alarm sistemlerinde kullanılan farklı modellerdeki doğal gaz veya LPG dedektörlerinin görselleri görülmektedir.



Görsel 4.37: Yangın alarm sistemlerinde kullanılan farklı modellerdeki gaz dedektörleri

Doğal gaz-LPG dedektörlerinin montajı esnasında dikkat edilmesi gerekenler

a) Dedektör tabanında bulunan montaj delikleri yardımıyla taban duvara sabitlenir.

b) Tesisat kabloları ilgili yerlere gevşek olmayacak şekilde bağlanır.

c) Dedektör gövdesi taban üzerine sadece bir şekilde yerleştirilebilmektedir. Dedektör gövdesi tabana yerleştirilip saat yönünde çevrilerek montaj yapılmalıdır.

ç) Dedektör gaz kaynağı ile aynı ortamda bulunmalıdır.

d) Dedektör fan, kapı vb. hava akımının güçlü olduğu yerlere yakın olmayan yere takılmalıdır.

e) Duman veya buhardan etkilenebileceği yerlere takılmamalıdır.

f) Patlayıcı gaz olan LPG havadan ağırdır, bu nedenle dedektör yerden 15-25 cm yüksekliğe ve LPG kaynağına yatay olarak 150 cm uzağa monte edilmelidir (Görsel 4.38).

g) Patlayıcı bir gaz olan doğal gaz havadan hafiftir, bu nedenle dedektör tavandan 5-15 cm aşağıya ve doğal gaz kaynağına yatay olarak 150 cm uzağa monte edilmelidir (Görsel 4.38).

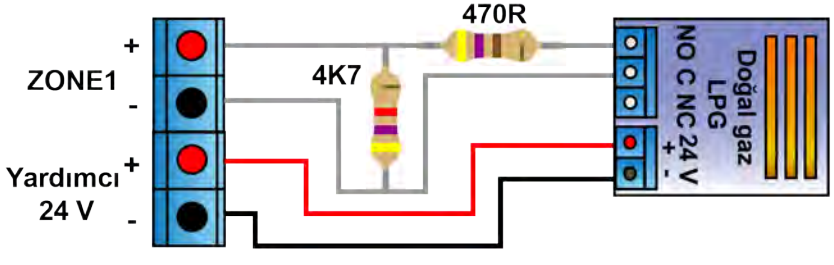


Görsel 4.38: Doğal gaz-LPG dedektör montajı

Alarm durumunda yapılması gerekenler

- Gaz tüketen cihazların vanası kapatılır.
- Kıvılcım çıkarmamak için pencereler yavaşça açılır.
- Patlamaya neden olmamak için ışıklar açılmaz, gazı dağıtmak için dahi olsa havalandırma fanı veya başka elektrikli cihazlar çalıştırılmaz ve kibrit kullanılmaz.
- Gaz şirketine veya itfaiyeye haber verilir.

Görsel 4.39'da doğal gaz-LPG dedektörünün bağlantısı görülmektedir.



Görsel 4.39: Doğal gaz-LPG dedektörünün bağlantısı

Duman, ısı, multi ve alev dedektörleri ikili yanmaz kablo ile santralin **zone** uçlarına bağlanırken doğal gaz-LPG dedektörü dört kablo ile bağlanır. Kabloların ikisi diğer dedektörlerde olduğu gibi data iken diğer ikisi besleme gerilimidir. Ayrıca doğal gaz-LPG dedektörü genellikle röle çıkışlı olarak bulunur. Röle, NO-C arası gaz algılandığı zaman kısa devre olacağı için **zone**a seri 470 Ω alarm direnci ile birlikte bağlanır. Eğer 470 Ω direnç bağlanmaz ise doğal gaz-LPG dedektörü gaz algıladığında NO-C kontağını kapasa bile alarm santrali bunu alarm sinyali olarak değil hata sinyali olarak algılar. Kuru kontak anahtarlama çıkışlı cihazların bağlantısının içinde alarm direnci yok ise mutlaka dışarıdan bağlanmalıdır.

4.2.10. Acil Çıkış Yönlendirme Levhaları

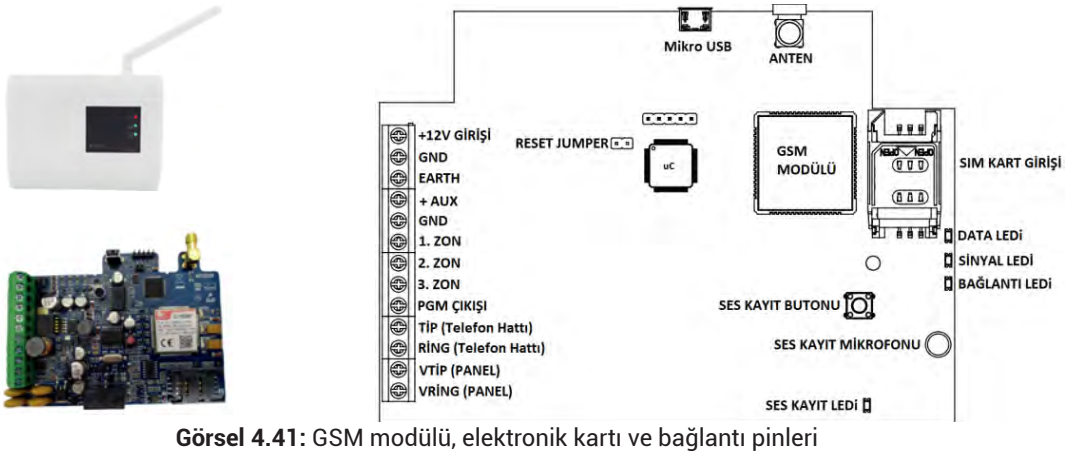
Yangın anında, birden fazla çıkışı olan bütün binalarda kullanıcıların çıkışlara kolaylıkla ulaşabilmesi için acil durum yönlendirme levhaları kullanılır. Yönlendirme levhalarının içindeki batarya (pil), 3-5 saat elektrik kesintilerinde ve acil durumlarda çalışmaya imkân sağlar. Görsel 4.40'ta acil çıkış yönlendirme levhaları görülmektedir.



Görsel 4.40: Acil çıkış yönlendirme levhaları

4.2.11. GSM Modülü

GSM modülü, yangın alarm panellerinin hafızasına önceden yüklenen sesli uyarıyı yangın anında kullanıcıya GSM üzerinden bildirmesi için kullanılır. Görsel 4.41'de GSM modülü, elektronik kartı ve bağlantı pinlerinin görselleri görülmektedir.



Görsel 4.41: GSM modülü, elektronik kartı ve bağlantı pinleri

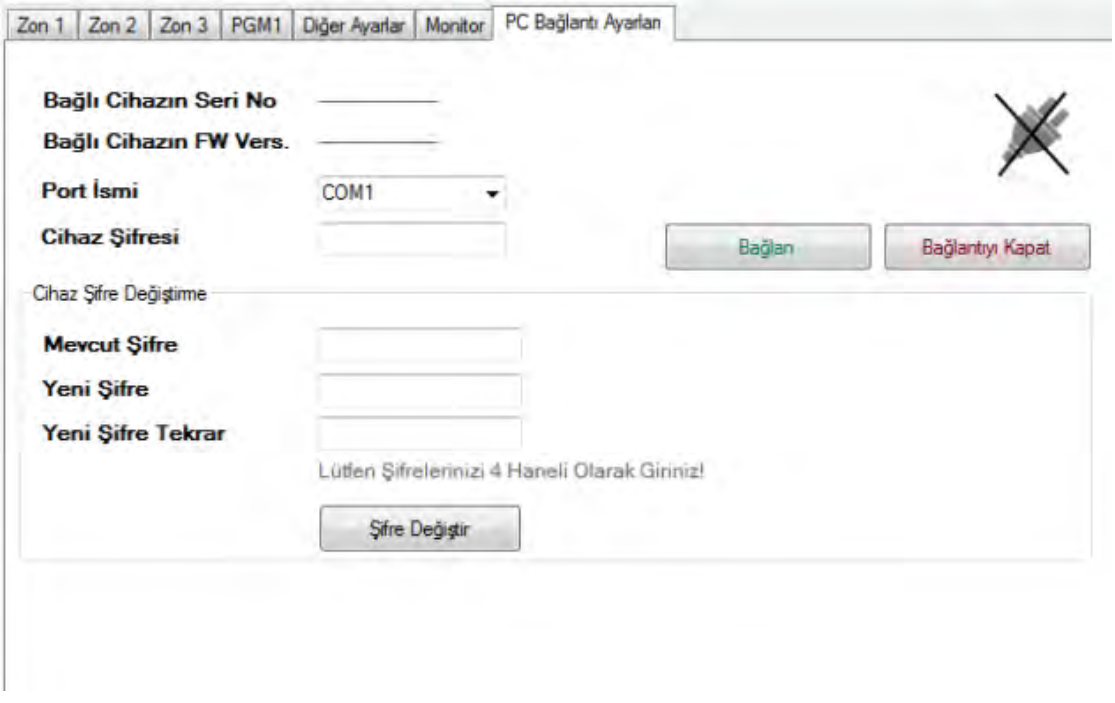
GSM modülü bağlantı pinlerinin işlevleri aşağıda verilmiştir.

1. +12V, dışarıdan (panelden) gelen besleme geriliminin (+) ucudur.

2. GND, dışarıdan (panelden) gelen besleme geriliminin (–) ucudur.

3. EARTH, koruma için toprak bağlantı girişidir.
4. +AUX, harici +12 V çıkışıdır.
5. GND, harici çıkış için (–), **zone**lar için ortak uçtur.
6. PGM, ayarlanabilir çıkış tetik ucudur.
7. Tip-Ring (Tel), sabit telefona giden uçlardır.
8. VTİP-VRİNG, alarm panelinin telefon girişine giden uçlardır.

GSM modülleri genellikle bir bilgisayar programı aracılığı ile programlanır. Bu program ile işlem yapabilmek için önce mikro USB soketi bilgisayarın USB soketine bağlanır. GSM modülü her ne kadar USB üzerinden bilgisayara bağlansa da kendi üzerinde USB/RS 232 çeviricisi vardır. Yani program ilk açıldığında GSM modülü ile haberleşilmesi ve program ara yüzündeki bilgilerin aktarılması için PC bağlantı sekmesinden önce COM port ayarı yapılır. Cihaz şifreleri girilerek de cihaza bağlantı yapılır (Görsel 4.42).

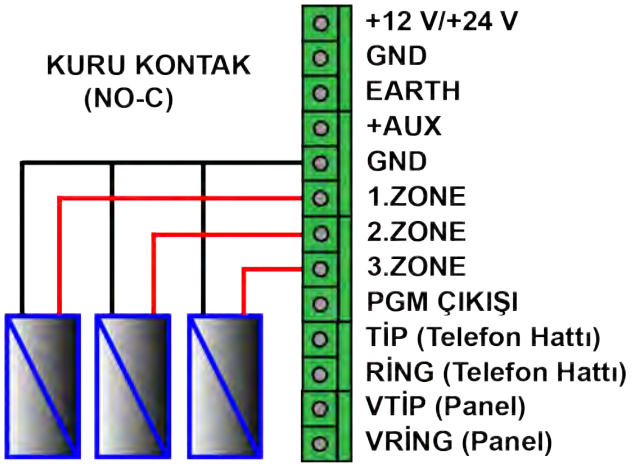


Görsel 4.42: GSM modülünün bilgisayar programının ekran görüntüsü

GSM modülü ile bilgisayar bağlantısı yapıldıktan sonra diğer ayarlar sekmesinden sinyal seviyesi, SMS numarası gibi GSM ayarları yapılır. **ZONE1, ZONE2** ve **ZONE3** sekmelerine ayrı ayrı sinyal geldiğinde aranacak telefon numaraları, SMS atılacak numaralar, **ZONE** alarmında gönderilecek mesaj, **ZONE** alarmının düzelmesinde gönderilecek mesaj girilir. PGM sekmesine SMS kabul edilecek telefon numarası, mesajı gibi bilgiler girilebilir. Bütün ayarlar bu program ara yüz ekranında yapıldıktan sonra "Ayarları Karta Yükle" sekmesi tıklanarak GSM modülü ayarlanmış olur.

GSM modülü program ayarı tamamlandıktan sonra besleme gerilimi (+12 V) verilir. 3 adet **ZONE** girişi ile ayrı ayrı GND arasında oluşacak kuru kontak anahtarlamasına göre hafızasındaki mesajı ve SMS bilgilerini ilgili telefon numaralarına gönderir (Görsel 4.43).

Alarm panelinin kullanılmadığı yerlerde herhangi bir sensörden gelen kuru kontak alarm durumunu GSM üzerinden iletebilir.



Görsel 4.43: GSM modülünün ZONE uçları ile GND arasındaki kuru kontak anahtarlama bağlantısı

4.2.12. Karbonmonoksit (CO) Alarm Santrali ve Gaz Dedektörü

Karbonmonoksit (CO) gazı, zehirli bir gazdır. Yangın veya yanma işleminin bir sonucu olarak ortaya çıkan ürünlerden biri olduğu gibi araba egzozlarından çıkan gazda karbonmonoksit olarak bilinir. Vücuda solunum yolu ile girer. Karbonmonoksit vücutta parçalanmaz, solunum yoluyla dışarı atılır. Karbonmonoksit zehirlenmeleri sıklıkla kapalı bir ortamda meydana gelen yanma sırasında olur.

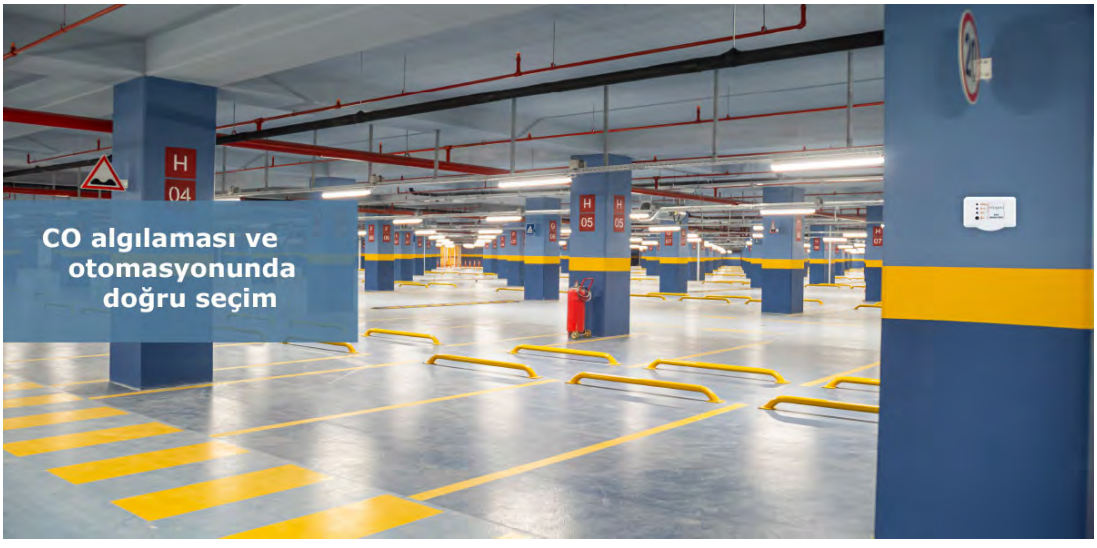
Karbonmonoksit dedektörleri genellikle AVM, hastane, bina vb.nin otoparklarında kullanılır. Bu dedektörler ortamdaki insan soluma düzeyine takılmalıdır. Bu düzey genellikle yerden 150 cm yüksekliktir. Karbonmonoksit zehirleyici bir gazdır ve havanın ağırlığına yakındır. Bu yüzden insan soluma yüksekliği hava ile homojen dağılacağından uygundur. Karbonmonoksit gaz dedektörünün monte edileceği yer seçilirken ortamda normal hava sirkülasyonunun engellenmediği bir yer seçilmelidir. Dedektör, yanlış alarmların önlenmesi için ocak veya sobaların yakınına değil, en az 1 metre uzağına konmalıdır.

Son zamanlarda karbonmonoksit gaz dedektörleri AVM otoparklarında yaygın olarak kullanıldığından ve AVM otoparklarına yüksek sayıda dedektör takıldığından ayrı alarm santralleri üretilmiştir. Görsel 4.44'te karbonmonoksit alarm santralleri ve dedektör görselleri görülmektedir. Karbonmonoksit alarm santralleri konvansiyonel sistem mantığında çalışmaktadır.



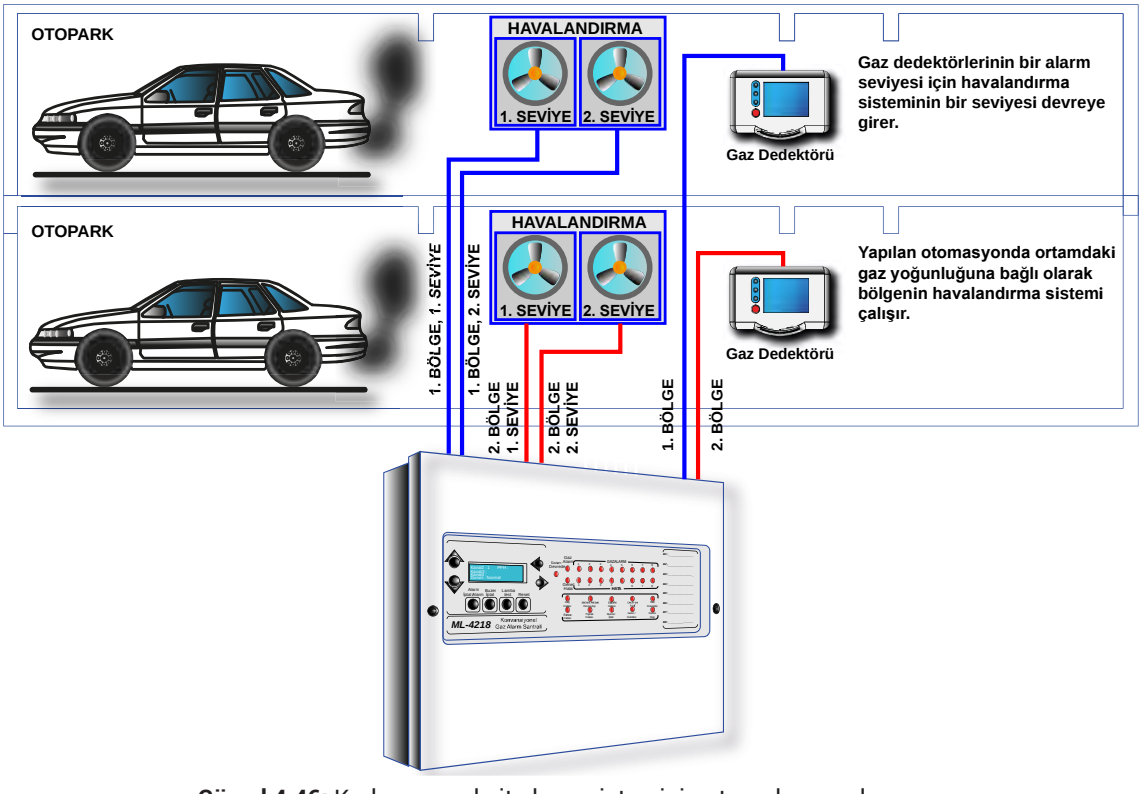
Görsel 4.44: Karbonmonoksit alarm paneli ve dedektörler

Görsel 4.45'te karbonmonoksit alarm sisteminin otoparka yerleşim görselleri, Görsel 4.46'da ise karbonmonoksit alarm sisteminin otopark uygulaması görülmektedir.



Görsel 4.45: Karbonmonoksit alarm sisteminin otoparka yerleşim resimleri

Otoparka kurulumu yapılan sistem ortamdaki karbonmonoksit (CO) gazı miktarını ölçer. CO gazı seviyesi dedektörler ile algılanır. CO gazı seviyesi eğer birinci seviyede ise birinci seviye havalandırma sistemi devreye girer. Eğer ikinci seviyede ise her iki seviye havalandırma sistemi devreye girer. Ortamdaki araç egzozlarından çıkan zehirli CO gazının boşaltılmasını sağlar.



Görsel 4.46: Karbonmonoksit alarm sisteminin otoparka uygulanması

4.2.13. Otomatik Yangın Söndürme Sistemleri

Yangın söndürme sistemleri, herhangi bir dikkatsizlik veya ihmal sonucu oluşan yangınları söndürmek için geliştirilmiştir. Genel olarak manuel (el ile) ve otomatik söndürme sistemleri olarak sınıflandırılabilir.

Manuel yangın söndürme sistemleri, genellikle el ile müdahale edilmek üzere dizayn edilmiştir (Görsel 4.47a). Kendi içinde sulu, gazlı ve köpüklü söndürme sistemleri olarak sınıflandırılır. Görsel 4.47b'de ise gazlı söndürücülerin görselleri görülmektedir.

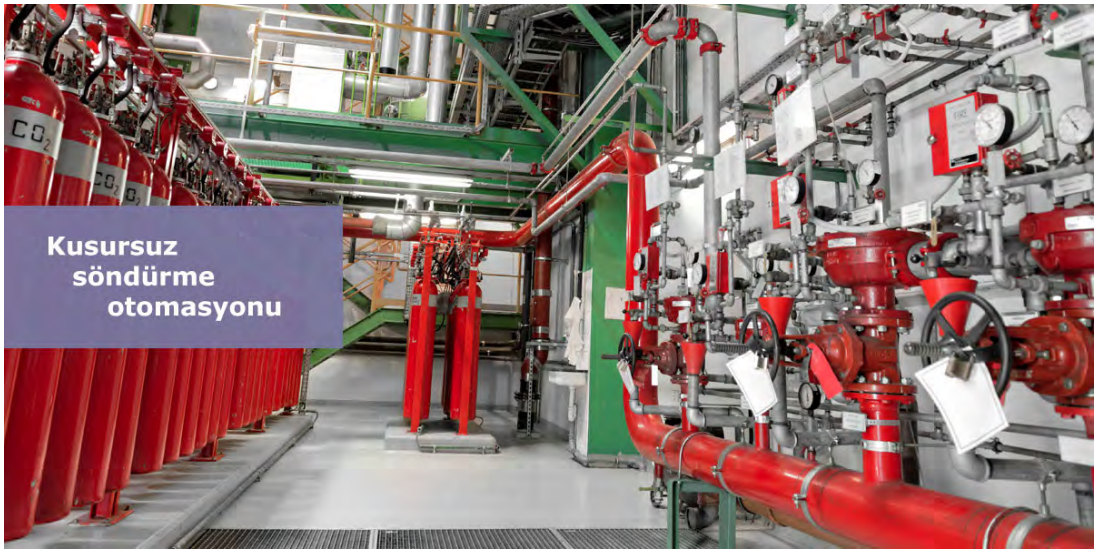


Görsel 4.47a: Yangın dolabı



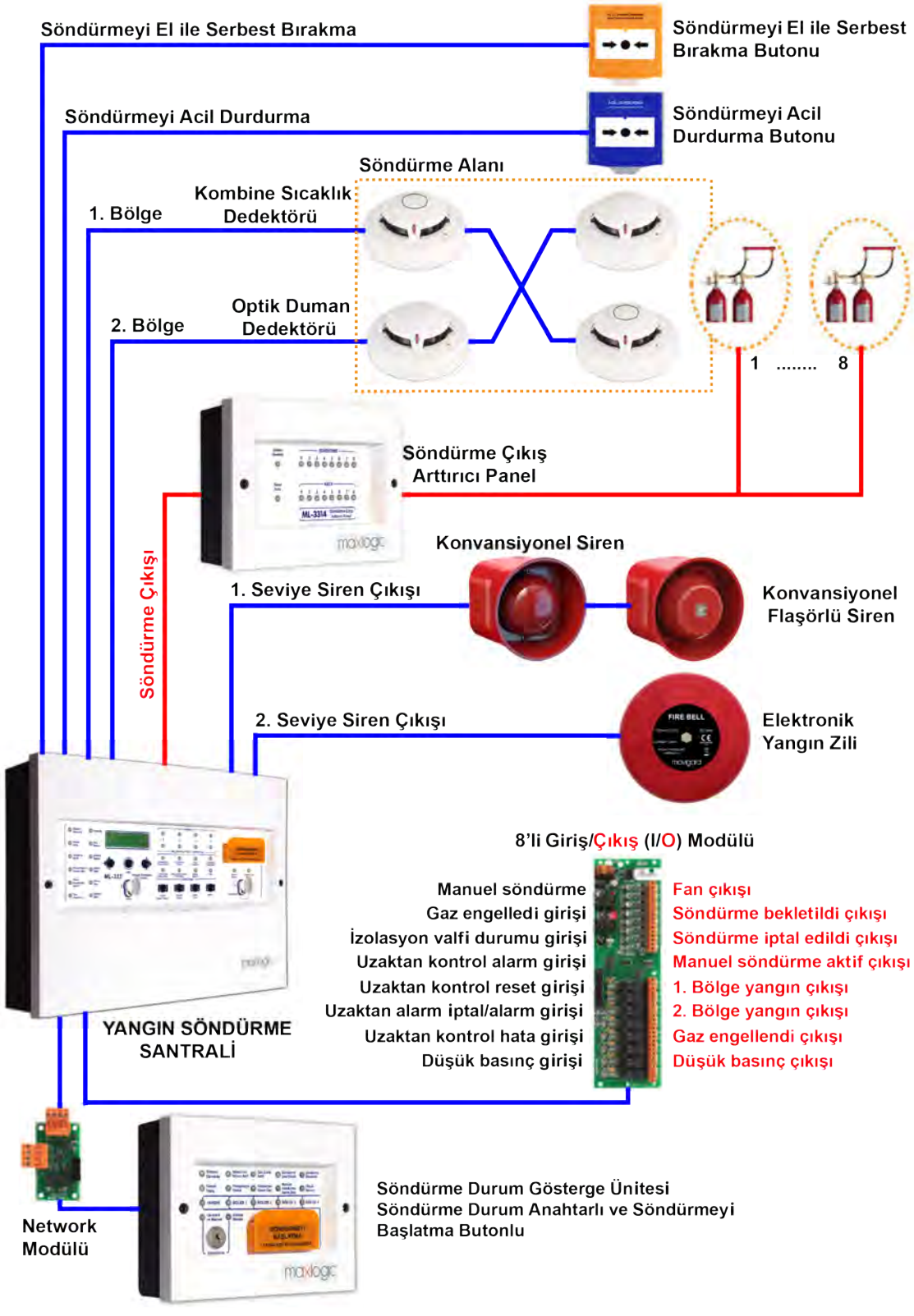
Görsel 4.47b: Gazlı söndürücüler

Otomatik yangın söndürme sistemleri, yangın olayına çabuk müdahale edilebilmesi ve oluşabilecek tehlikelerin en kısa zamanda önlenebilmesi için geliştirilen elektronik kontrollü ürünlerdir. Son zamanlarda otomatik yangın söndürme cihazlarının kontrolü için yangın söndürme santralleri üretilmiştir. Üretilen yangın söndürme santralleri de konvansiyonel sistem mantığında çalışmaktadır. Görsel 4.48'de otomatik yangın söndürmede kullanılan ürünlerin görselleri görülmektedir.



Görsel 4.48: Otomatik yangın söndürmede kullanılan ürünler

Görsel 4.49'da otomatik yangın söndürme sisteminin uygulanma şeması görülmektedir. Yangın söndürme santrali dedektörlerden gelen yangın sinyallerine göre söndürme ürünlerini, sesli ve ışıklı uyarı cihazlarını harekete geçirir. Söndürme santrali aynı zamanda ihbar butonlarından gelen sinyallere göre söndürme sistemini durdurma ve serbest bırakma işlevlerini de yerine getirir. Yangın alarm santrallerinde ihbar butonları "Yangın var." bilgisini gönderirken otomatik söndürme santrallerinde ise söndürme işlemini durdurma bilgisini göndermektedir.



Görsel 4.49: Otomatik yangın söndürme sisteminin uygulanma şeması

4.2.14. Yangın Alarm Sistemlerinde Kullanılan Yardımcı Ürünler

Adreslenebilir sistemin konvansiyonel sistemden farklılıkları vardır. Bunun için konvansiyonel sistemde kullanılan ürünleri adreslenebilir sisteme dâhil etmek veya adreslenebilir sistemi daha güvenli kullanabilmek için ekstra ürünler kullanılır.

4.2.14.1. Adresli Siren Kontrol Modülü

Yangın senaryosuna uygun olarak sesli ve flaşörlü uyarı cihazlarını kontrol eder. Zaman gecikmesine olanak sağlar. Kolay monte edilebilen, adresli elektronik kontrol modülüdür. Her modül bir adres almaktadır. **Loop** hattına ilave olarak extra 24 V DC besleme enerjisine ihtiyaç duyar. Görsel 4.50'de adresli siren kontrol modülleri görülmektedir.



Görsel 4.50: Adresli siren kontrol modülleri

4.2.14.2. Adresli Röle Modülü

Giriş/çıkış modülleri, çeşitli algılama ve alarm cihazları için bir neden ve etki planı kurmayı sağlar ve yangın güvenliğinde yer alan acil durum önlemlerini aktif hâle getirir. Modüller; yangın algılama senaryosundaki ısıtma, havalandırma, yangın kapıları, asansör vb. sistemlerdeki haricî cihazlar arasında bir arayüz olarak hareket eder. Röle modülü, giriş terminalleri ile algılama cihazlarını izlemede kullanılabilir ve iki ayrı cihazı izleyebilir. Ayrıca yangın damperleri, havalandırma, iklimlendirme ekipmanı gibi yangın kontrol mekanizmalarını çıkış terminalleriyle aktif hâle getirebilir. Role modülü cihazın alarm, kısa-açık ve topraklama durumlarını supervize edebilir. Bu cihazlar aynı yangın algılama **loop**undan çalıştırılır ve bir adres işgal eder. Ayrıca 24 V harici beslemeye ihtiyaç duyar. Görsel 4.51'de adresli röle modülleri görülmektedir.



Görsel 4.51: Adresli röle modülleri

4.2.14.3. Adresli Kontak İzleme Modülü

Giriş modülleri çeşitli algılama ve alarm cihazları için bir neden planı kurmayı sağlar. Modüller yangın algılama senaryosundaki izleme yapılabilecek mekanik aksamların (splinkler kesme vanaları, splinkler akış anahtarları vb.) arasında bir arayüz olarak hareket eder. İzleme modülü giriş terminalleri ile algılama cihazlarını izlemede kullanılır. İzleme modülü cihazın alarm, kısa-açık ve topraklama durumlarını supervize edebilir ve kontrol paneline anında bildirir. Bu cihazlar aynı yangın algılama **loop**undan çalıştırılır ve bir adres işgal eder. Haricî beslemeye ihtiyaç duymaz.

Görsel 4.52'de adresli **kontak** izleme modülleri görülmektedir.



Görsel 4.52: Adresli kontak izleme modülleri

4.2.14.4. Adresli Zone Modülü

Zone modülü adresli yangın alarm sistemlerinde konvansiyonel algılama cihazlarının bulunduğu bölgeleri tek adres altında toplar. Tüm konvansiyonel algılama cihazları ile entegre çalışır. **Zone** modülü cihazın alarm, kısa-açık ve topraklama durumlarını supervize edebilir ve kontrol paneline anında bildirir. Bu cihazlar aynı yangın algılama lopundan çalıştırılır ve bir adres işgal eder. Haricî beslemeye ihtiyaç duyar. Görsel 4.53'te adresli **zone** modülleri görülmektedir.



Görsel 4.53: Adresli zone modülleri

4.2.14.5. İzolatör Modülü

İzolatör, kısa devre koruma modülüdür. Bu modüller **loop**ta oluşan kısa devre bölgesini **loop**tan ayırarak **loop** hattını ve **loop** hattındaki cihazları korur. Bu ürün doğrudan **loop** beslemeli olup bir sınırlama olmadan istenen miktarda kullanılabilir. Herhangi bir adres işgal etmez. İki izolatör arasındaki **loop** hattında kısa devre oluştuğunda diğer hattın çalışmasına olanak sağlar. Arıza durumunda üzerinde bulunan hata LED'i sürekli yanar. Görsel 4.54'te izolatör modülleri görülmektedir.



Görsel 4.54: İzolatör modülleri

4.2.14.6. Termal Printer

55 mm, 32 karakter termal **printer**dır. Panele opsiyonel olarak takılabilen bu printer sayesinde oluşan alarm, hata ve olaylar yazdırabilmektedir. Bu sayede paneldeki geçmiş olaylar çıktı olarak alınıp arşivlenebilmektedir. Termal kâğıt bittiğinde değiştirilebilir. Kesme bıçaklıdır. Görsel 4.55'te termal printer görselleri görülmektedir.



Görsel 4.55: Termal printer

4.2.14.7. Network Haberleşme Kartı

Yangın alarm panellerinin birbirleri ve GUI yazılımı ile haberleşmesini sağlar. Bağlantı arayüzü olarak RS-485, ethernet, modbus, GPRS kullanılabilen modellerine sahiptir. Bu ürün opsiyonel olup bu tür haberleşme talep edildiğinde panel içindeki bağlantı noktalarına bağlanarak kullanılabilmektedir. Görsel 4.56'da network haberleşme kartı ve modülü görülmektedir.





Görsel 4.56: Network haberleşme kartları

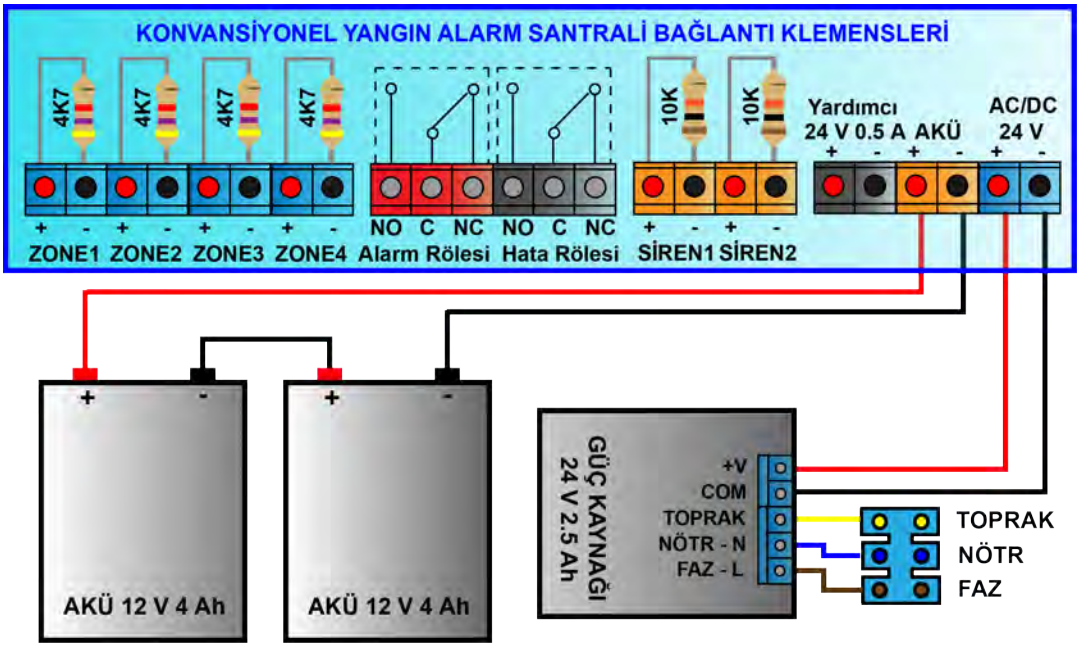
Uygulama Adı: Konvansiyonel Yangın Alarm Santralinin Testi No.: 1

Amaç: **Konvansiyonel yangın alarm santralinin testini yapmak.**

Görsel 4.57'de Konvansiyonel yangın alarm santrali bağlantı klemensleri görülmektedir. Santral ilk enerjilendiğinde bütün LED'ler yanar ve yaklaşık 4 sn. sonra normal çalışma durumuna geçer. Bu durumda sadece sistem devrede LED'i yanar. Herhangi bir **zone**da açık devre, kısa devre, hat sonu direnci hatası veya dedektör çıkarılması durumunda o **zone**a ait hata LED'i ve genel hata LED'i yanar, **buzzer** çalar ve hata rölesi aktif olur. Hata durumu ortadan kalkınca sistem otomatik olarak normal çalışma durumuna geçer.

Herhangi bir bölgede yangın alarm durumu oluştuğunda yangın LED'i yanar, **buzzer** ve sirenler çalar, yangın rölesi aktif olur. Bu durumda kilit açık duruma getirilerek alarm iptal butonuna basılarak sirenler susturulur ve alarm veren **zone** kontrol edilir. Gerçekten yangın varsa "ALARM ON" butonuna basılarak sirenler aktif hâle getirilir ve tahliye işlemi başlatılır. Yangın yoksa "RESET" butonuna basılarak sistem normal çalışma konumuna getirilir.

Siren 1 açık devre, siren 1 kısa devre, siren 2 açık devre, siren 2 kısa devre, güç hatası, akü hatası, 24 V DC hatası ve toprak hatası durumlarında ise ilgili hata LED'i ve genel hata LED'i yanar, **buzzer** çalar ve hata rölesi aktif olur. Hatanın ortadan kalkması durumunda ise sistem otomatik olarak normal çalışma durumuna geçer.



Görsel 4.57: Konvansiyonel yangın alarm santrali bağlantı klemensleri

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak konvansiyonel yangın alarm santralinin kapağını açınız. **Zone** uçlarında hat sonu direnci (4,7 kΩ, 6,8 kΩ) olduğunu, siren uçlarında 10 kΩ hat sonu direnci olduğunu gözlemleyiniz. Eksik ise tamamlayınız.
2. Öğretmeninizin kontrolünde konvansiyonel yangın alarm santraline şehir şebekesi ve akü gerilimlerini vermek için klemens bağlantılarını yapınız. **RESET** butonuna basarak santrali resetleyiniz.

3. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
4. Akünün (+) bağlantısını sökünüz ve akü hata LED'inin yandığını gözlemleyiniz. Sonra akünün (+) ucunu tekrar ilgili klemense bağlayınız. Akü hata LED'inin söndüğünü gözlemleyiniz.
5. Şehir şebekesini kesiniz. Güç hata lambasının yandığını gözlemleyiniz. Sonra şehir şebekesini tekrar ilgili klemense bağlayınız. Güç hata lambasının söndüğünü gözlemleyiniz.
6. **Alarm ON** butonuna basınız. **Buzzer**ın çaldığını ve yangın LED'inin yandığını gözlemleyiniz.
7. **Buzzer OFF** butonuna basarak **buzzer**ın sustuğunu teyit ediniz.
8. **Alarm OFF** butonuna basınız. **Buzzer**ın kesik kesik çaldığını ve yangın LED'inin söndüğünü gözlemleyiniz.
9. **RESET** butonuna basarak yangın alarm santralini yeniden başlatınız.
10. Ölçü aletini ohmmetre ya da **buzzer** kademesine alıp hata rölesinin NO-C ile NC-C arasını ölçünüz. Sonucu Tablo 4.1'de ilgili hücreye yazınız.
11. **ZONE1**'deki hat sonu direncini sökünüz. **ZONE1** hata LED'inin yandığını ve **buzzer**ın çaldığını gözlemleyiniz. Yine ölçü aleti ile hata rölesinin NO-C ve NC-C arasını ölçünüz. Sonucu Tablo 4.1'de ilgili hücreye yazınız.
12. **ZONE1**'deki hat sonu direncini yerine takınız. **ZONE1** hata LED'inin söndüğünü ve **buzzer**ın sustuğunu gözlemleyiniz.
13. Ölçü aletini ohmmetre ya da **buzzer** kademesine alıp yangın alarm rölesinin NO-C ile NC-C arasını ölçünüz. Sonucu Tablo 4.1'de ilgili hücreye yazınız.
14. **ZONE1**'deki hat sonu direncini sökünüz ve yerine 470 Ω direnç bağlayınız. **ZONE1** yangın LED'inin yandığını ve **buzzer**ın çaldığını gözlemleyiniz. Yine ölçü aleti ile yangın alarm rölesinin NO-C ve NC-C arasını ölçünüz. Sonucu Tablo 4.1'de ilgili hücreye yazınız.
15. **ZONE1**'e taktığınız 470 Ω direnci söküp yerine hat sonu direncini tekrar takınız. "RESET" butonuna basınız. Santralin normal moda döndüğünü gözlemleyiniz.
16. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Tablo 4.1: Alarm ve Hata Rölelerinin Ölçüm Sonuçları

Ölçülecek Röle	Hata Modu	NO-C (Ölçülen R,Ω)	NC-C (Ölçülen R,Ω)
Hata Rölesi	Pasif		
	Aktif		
Alarm Rölesi	Pasif		
	Aktif		

Sorular

1. Hat sonu direnci **zone**da takılıyken neden sistem normal çalışmaktadır? Açıklayınız.
2. Hat sonu direnci **zone**da takılı değilken neden sistem hata vermektedir? Açıklayınız.
3. Hat sonu direnci (4,7 kΩ, 6,8 kΩ) yerine 470 Ω (680 Ω) direnç takınca santral neden yangın moduna geçmiştir? Açıklayınız.
4. Hata rölesinin çalışmasını Tablo 4.1'e göre açıklayınız.
5. Yangın rölesinin çalışmasını Tablo 4.1'e göre açıklayınız.

Uygulama Adı: Konvansiyonel Yangın Alarm Santralinde Ölçüm No.: 2

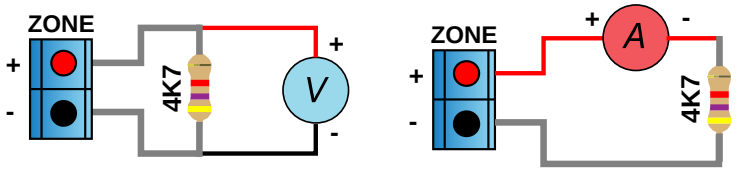
Amaç: Konvansiyonel yangın alarm santralinde ölçüm yapmak.

Konvansiyonel yangın alarm santralleri **zone**lardan çekilen akıma göre çalışmaktadır. Burada da en önemli görev hat sonu direncine düşmektedir. Yangın alarm santralinde **zone**da hat sonu direnci (47 kΩ, 6,8 kΩ, 10 kΩ) varken **zone**dan denge akımı çekiliyor ise yangın alarm santrali denge modunda çalışır. **Zone**da hat sonu direnci (470 Ω, 680 Ω, 1 kΩ) varken **zone**dan alarm akımı çekiliyor ise yangın alarm santrali alarm modunda çalışır. **Zone**da açık devre veya kısa devre var iken **zone**dan akım çekilmiyor ya da aşırı akım çekiliyor ise yangın alarm santrali hata modunda çalışır.

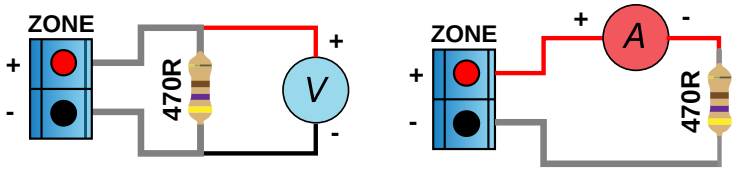
Konvansiyonel yangın alarm santrallerinde hat sonu direnci genellikle **zone**larda 4,7 kΩ, siren çıkışlarında ise 10 kΩ olarak kullanılır. 4,7 kΩ ve 10 kΩ direnç değerleri farklı marka modellerde 4,7 kΩ, 6,8 kΩ, 10 kΩ gibi değerler olabilir. Ayrıca konvansiyonel yangın alarm santrallerinde kullanılan **zone** alarm direnci de genellikle 470 Ω'dur. Fakat bu değer de 470 Ω, 680 Ω, 1 kΩ gibi değerler olabilir. Temrinlerdeki görsellerde bu üç değer de kullanılacaktır. Fakat sizin elinizdeki ürünün marka ve modeline göre uygun olan hat sonu dirençlerini kullanınız. Bu değerleri ürünün kullanım kılavuzlarında bulabilirsiniz. Ürünü sıfır aldıysanız zaten üzerinde hat sonu dirençleri takılı olarak gelecektir.

Konvansiyonel yangın alarm santrallerinde 500 m'ye kadar ST 0,80 yanmaz kablo, 500 m üstündeki mesafelerde ise ST 1,5 yanmaz kablo kullanılır. Kullanılan kablonun uzunluğuna göre kablonun iç direnci 100 Ω'u geçmez. Hat sonu direnci 4,7 kΩ değerinde yüksek bir değer olduğu için 4,7 kΩ=4700 Ω yanında 100 Ω'luk fazla ya da eksik direnç değeri olması bu gibi sistemlerde yapılan ölçümlerde çok fazla bir farklılık oluşturmaz. Yangına dayanıklı olması ve ek yapılmaması önemlidir.

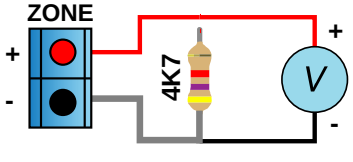
Görsel 4.58a'da **DENGE**, Görsel 4.58b'de **ALARM**, Görsel 4.58c'de ise **HATA** modlarında **zone**dan akım ve gerilim ölçme devreleri görülmektedir.



Görsel 4.58a: DENGE modunda **ZONE**'dan akım ve gerilim ölçme devresi



Görsel 4.58b: ALARM modunda **ZONE**'dan akım ve gerilim ölçme devresi



Görsel 4.58c: HATA modunda **ZONE**'dan gerilim ölçme devresi

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak konvansiyonel yangın alarm santralinin kapağını açınız. **Zone** uçlarında hat sonu direnci (4,7 kΩ, 6,8 kΩ) olduğunu, siren uçlarında 10 kΩ hat sonu direnci olduğunu gözlemleyiniz. Eksik ise tamamlayınız.
2. Öğretmeninizin kontrolünde konvansiyonel yangın alarm santraline şehir şebekesi ve akü gerilimlerini vermek için bağlantılarını yapınız. **RESET** butonuna basarak santrali resetleyiniz.
3. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
4. Tablo 4.2'yi doldurunuz.
5. Görsel 4.58a'daki devre şemalarını yangın alarm santralinin istediğiniz **zone** çıkışında uygulayarak sonuçları Tablo 4.3'e yazınız.
6. Görsel 4.58b'deki devre şemalarını yangın alarm santralinin istediğiniz **zone** çıkışında uygulayarak sonuçları Tablo 4.3'e yazınız.
7. Görsel 4.58c'deki devre şemalarını yangın alarm santralinin istediğiniz **zone** çıkışında uygulayarak sonuçları Tablo 4.3'e yazınız.
8. **ZONE1**'e 4,7 kΩ hat sonu direncini yerine takınız. **RESET** butonuna basınız. Santralin normal moda döndüğünü gözlemleyiniz.
9. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Tablo 4.2: Akü ve Yardımcı 24 V Gerilim Çıkışının Ölçüm Sonuçları

Şehir Şebekesi	Akü Durumu	Akü Soketi (V)	Akü Gerilimi (V)	Yardımcı 24 V (V)
220V var iken	Santrale bağlı			
	Bağlı değil			
220V yok iken	Santrale bağlı			
	Bağlı değil			

Tablo 4.3: Yangın Alarm Santralinin Çalışma Modlarındaki Zone Gerilim ve Akım Ölçüm Sonuçları

Alarm Santralinin Çalışma Modu	ZONE No.	Hat Sonu Direnç Değeri	Santralin Durumu İstenilen Çalışma Modunda	ZONE Gerilimi (V)	ZONE Akımı (A)
DENGE Modu		4,7 kΩ	Çalışıyor / Çalışmıyor		
		6,8 kΩ	Çalışıyor / Çalışmıyor		
		10 kΩ	Çalışıyor / Çalışmıyor		
ALARM Modu		470 Ω	Çalışıyor / Çalışmıyor		
		680 Ω	Çalışıyor / Çalışmıyor		
		1 kΩ	Çalışıyor / Çalışmıyor		
HATA Modu		Direnç yok	Çalışıyor / Çalışmıyor		

Sorular

1. Tablo 4.3'teki sonuçlara göre hat sonu direncinin önemini yangın alarm santralinin çalışma modlarını dikkate alarak açıklayınız.
2. Yangın alarm sistemlerinde kullanılan kablonun uzunluğu neden önemli değildir? Belirtiniz.
3. Güç kaynağı **switchmode** olan 24 V 4,5 A yangın alarm santrallerinde 2 tane seri bağlı 12 V 4 A aküyü şarj edecek gerilim değeri nasıl elde edilir? Araştırınız.

NOT DEFTERİ

Uygulama Adı: Konvansiyonel Yangın Alarm Santraline Dedektör Bağlantısı No.: 3

Amaç: Konvansiyonel yangın alarm santraline dedektör bağlantısı yapmak.

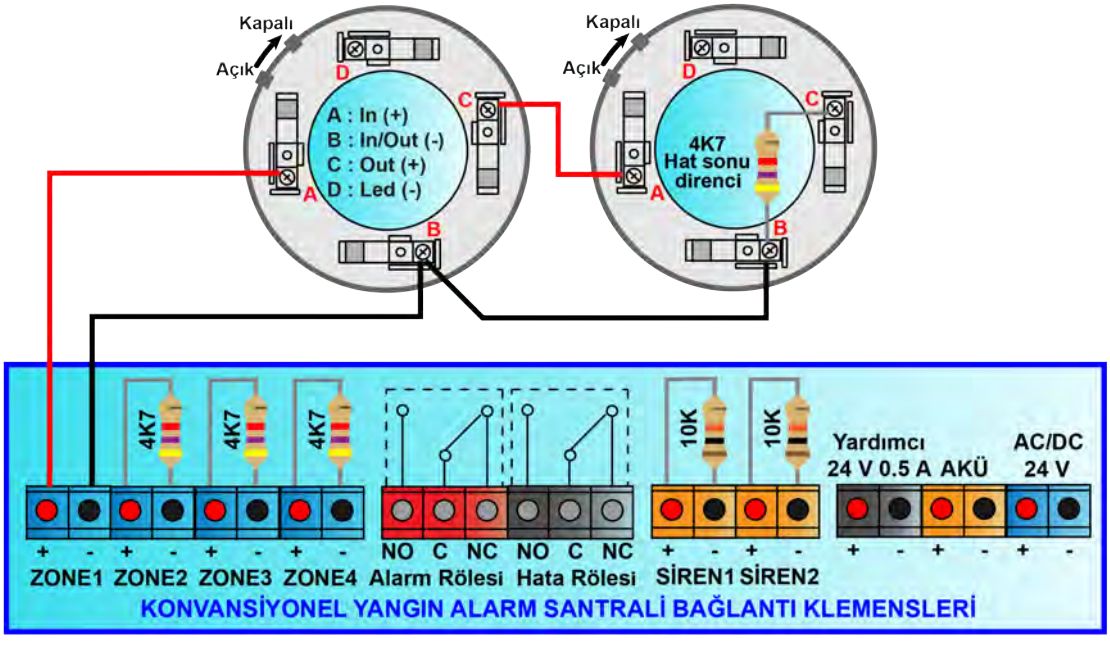
Konvansiyonel yangın alarm santraline dedektör bağlantısı yapılırken en önemli kısım dedektör tabanının montajı ve tabana bağlanacak olan kabloların hangi numaralara bağlanacağıdır. Bu işlem, farklı marka ve modellerde çeşitlilik gösterse de her modelin dedektör gövdesinin tabanında bulunan bağlantı etiketinin üzerindeki numaralara dikkat edilerek yapılır (Görsel 4.59).

Genel bağlantı mantığı siyah (-,eksi) kablo santraldeki **zone**- ucundan çıkar ve bütün dedektörlerin ilgili ayağına –COM (-ORTAK) girer ve çıkar. Kırmızı (+,artı) kablo ise santraldeki **zone+** ucundan çıkar ve 1. dedektörün +IN (+giriş, Z+IN) ucuna girer. Bir sonraki dedektöre 1. dedektörün +OUT (+çıkış, Z+OUT) ucundan çıkar ve 2. dedektörün +IN ucuna bağlanır. Kısaca kırmızı (+) kabloyu dedektör kontaklarında kapalı seri anahtar üzerinden geçiyor gibi düşünebilirsiniz. Bu şekilde bir **zone**a 20 adet dedektör bağlanabilir. En son dedektörün +OUT ucu ile –COM ucuna hat sonu direnci (4,7 kΩ, 6,8 kΩ gibi) takılarak **zone** bölge bağlantısı sonlandırılır.

Hat sonu direnci zaten konvansiyonel alarm panelinde zone üzerinde bulunur. İlgili **zone**a kablo bağlantıları yapılırken sökülür ve en son dedektöre takılır.

Konvansiyonel yangın alarm santrallerinde dedektör bağlantılarının numaralarını ve isimlerini elinizdeki ürünün marka ve modeline uygun olan bağlantısını seçerek kullanınız. Bu değerleri ürünün kullanım kılavuzlarında bulabilirsiniz.

Dedektörlerin taban ayağı montajı ve kablo bağlantıları bittikten sonra dedektörün gövdesi üzerine oturtulur. Dedektör gövdesinin duman, ısı, multi veya alev dedektörü olması önemli değildir. İhtiyacınıza göre istediğinizi takabilirsiniz.



Görsel 4.59: Konvansiyonel yangın alarm santralinde dedektör bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak konvansiyonel yangın alarm santralinin kapağını açınız. **Zone** uçlarında hat sonu direnci (4,7kΩ, 6,8 kΩ) olduğunu, siren uçlarında 10 kΩ hat sonu direnci olduğunu gözlemleyiniz. Eksik ise tamamlayınız.
2. Konvansiyonel yangın alarm santraline bağlantısını yapmak üzere plançeteye 2 adet dedektör taban ayağının montajını yapınız.
3. **ZONE1**'deki hat sonu direncini söküp kaybetmeyeceğiniz bir yere koyunuz.
4. Duman dedektörünün gövdesinin altında yazan numaraları takip ederek taban ayağının **ZONE 1**'e bağlantısını yapınız.
5. 2. dedektör bağlantısını yapınız ve hat sonu direncini takınız.
6. Yaptığınız dedektör bağlantısını Görsel 4.59'a ve dedektör gövdesinin altına göre kontrol ederek dedektör gövdelerini taban ayaklarına oturtunuz.
7. Akü bağlantısını kontrol ederek santrale şebeke gerilimini veriniz ve RESET tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
8. 1. dedektörün gövdesini yerinden çıkararak santralden hata sinyalini aldığınızı görünüz.
9. Dedektör gövdesini yerine takınız ve hata sinyallerinin pasif olduğunu gözlemleyiniz.
10. Dedektörünüze algıladığı unsuru; duman dedektörüne kâğıt yakıp dumanını yaklaştırınız, ısı dedektörünün sensör ucunu ısıtınız, alev dedektörü ise ateş yaklaştırınız. Yangın alarm santralinin yangın sinyali verdiğini gözlemleyiniz.
11. **BuzzerOFF** butonuna basarak **buzzer**ın sustuğunu teyit ediniz.
12. Dedektörün algıladığı unsuru dedektörden uzaklaştırarak santral üzerindeki **RESET** butonuna basarak yangın alarm santralini yeniden başlatınız.
13. Sistemin normal çalışma moduna döndüğünü gözlemleyiniz.
14. Sistemin akü bağlantılarını söküp şebeke gerilimini kesiniz.
15. Aynı işlemleri 6. işlem basamağından itibaren ısı, multi ve alev dedektörü için de gerçekleştiriniz.
16. Görsel 4.59'daki devre şemasını üç dedektör bağlanacak şekilde organize edip plançete üzerine eklentileri yapıp işlem basamaklarını baştan başlayarak tekrarlayınız.
17. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Sorular

1. 8. işlem basamağının özelliği nedir? Açıklayınız.
2. 8. işlem basamağı sırasında hata rölesinin çalışma prensibini açıklayınız.
3. 10. işlem basamağının özelliği nedir? Açıklayınız.
4. 10. işlem basamağı sırasında yangın alarm rölesinin çalışma prensibini açıklayınız
5. Görsel 4.59'daki devre şemasında hat sonu direnci takılmazsa devre nasıl çalışır? Açıklayınız.
6. Görsel 4.59'daki devre şemasında 2 No.lu **zone**a dört dedektör bağlanacak şekilde yeniden ekleyerek bağlantı şemasını çiziniz.

Uygulama Adı: Konvansiyonel Yangın Alarm Santralinde Kullanılan Dedektörlerin Akım ve Gerilimlerinin Ölçülmesi

No.: 4

Amaç: Konvansiyonel yangın alarm santralinde kullanılan dedektörlerin akım ve gerilimlerini ölçmek.

Konvansiyonel yangın alarm sistemlerinde kullanılan elemanlardan en önemlisi dedektörlerdir. Dedektörlerin, yangın alarm santralinin denge, hata ve alarm çalışma modlarındaki **zone**lardan çektiği akım ve **zone** uçlarında düşen gerilim değerlerinin ölçümü yapılacaktır.

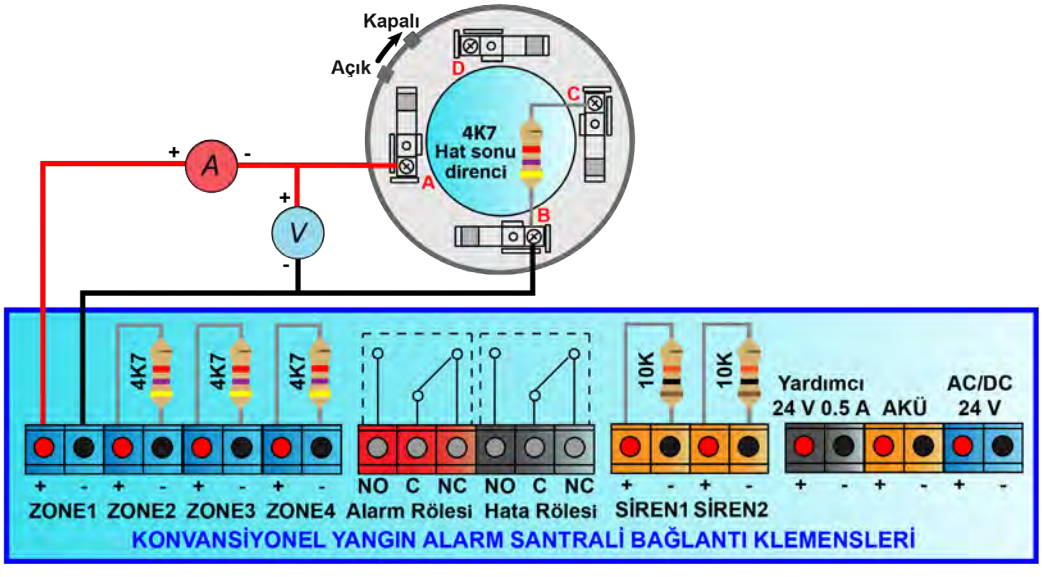
Dedektör taban ayağının montajı ve bağlantıları yapıldıktan sonra;

a) Denge modunda dedektör gövdesi taban ayağına monte edilmiştir. Dedektör algılayacağı unsuru beklemektedir.

b) Alarm modunda ise dedektör gövdesi taban ayağına monte edilmiştir. Dedektör algılayacağı unsuru algılamıştır. Yangın alarm santraline "Yangın bilgisi algılandı." mesajını verir.

c) Hata modunda ise dedektör gövdesi taban ayağına monte edilmiştir.

İşlem basamaklarını uygularken yangın alarm santrali alarm moduna geçer ise sistemi **RESET** butonuna basarak resetleyiniz. Hata moduna geçer ise ilgili hatayı düzelttiğinizde sistem normale dönecektir. Görsel 4.60'ta konvansiyonel yangın alarm santralinde dedektörlerin ölçüm bağlantıları görülmektedir.



Görsel 4.60: Konvansiyonel yangın alarm santralinde dedektörlerin ölçüm bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak konvansiyonel yangın alarm santralinin kapağını açınız. **ZONE** uçlarında hat sonu direnci (4,7 kΩ, 6,8 kΩ) olduğunu, siren uçlarında hat sonu direnci (10 kΩ) olduğunu gözlemleyiniz. Eksik ise tamamlayınız.
2. Konvansiyonel yangın alarm santraline bağlantısını yapmak üzere plançeteye bir adet dedektör taban ayağının montajını yapınız.
3. **ZONE1**'deki hat sonu direncini söküp kaybetmeyeceğiniz bir yere koyunuz.
4. Dedektörün gövdesinin altında yazan numaraları takip ederek taban ayağının, ilgili ölçü alet-

leriyle birlikte **ZONE1**'e bağlantısını yapınız.

5. Yaptığınız dedektör bağlantısını görsel 4.60'a göre kontrol ederek dedektör gövdesini taban ayaklarına oturtunuz.
6. Akü bağlantısını kontrol ederek santrale şebeke gerilimini veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
7. Tablo 4.4'ü doldurunuz.
8. **ZONE1**'e 4,7 kΩ hat sonu direncini yerine takınız. **RESET** butonuna basınız. Santralin normal moda döndüğünü gözlemleyiniz.
9. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Tablo 4.4: Yangın Alarm Santralinin Çalışma Modlarındaki Zone Gerilim ve Akım Ölçüm Sonuçları

Alarm Santralinin Çalışma Modu	ZONE No.	Dedektör Gövdesinin İsmi	Santralin Durumu İstenen Çalışma Modunda	ZONE Gerilimi (V)	ZONE Akımı (A)
DENGE Modu		Duman	Çalışıyor / Çalışmıyor		
		Isı	Çalışıyor / Çalışmıyor		
		Multi	Çalışıyor / Çalışmıyor		
		Alev	Çalışıyor / Çalışmıyor		
ALARM Modu		Duman	Çalışıyor / Çalışmıyor		
		Isı	Çalışıyor / Çalışmıyor		
		Multi	Çalışıyor / Çalışmıyor		
		Alev	Çalışıyor / Çalışmıyor		
HATA Modu		Dedektör gövdesi takılı değil	Çalışıyor / Çalışmıyor		

Sorular

1. Tablo 4.4'teki dedektörleri çalışma modlarındaki ölçülen gerilim ve akım değerlerine göre karşılaştırınız.
2. Tablo 4.4'teki sonuçlar ile Tablo 4.3'teki sonuçları yangın alarm santralinin çalışma modlarını, kullanılan dedektörleri ve ölçüm sonuçlarını dikkate alarak açıklayınız.

Uygulama Adı: Konvansiyonel Yangın Alarm Santraline Yangın İhbar Butonu Bağlanması No.: 5

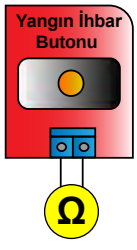
Amaç: Konvansiyonel yangın alarm santraline yangın ihbar butonu bağlamak.

Yangın ihbar butonu, yangın alarm santraline el ile "Yangın var." bilgisini göndermek için kullanılır. Yangın ihbar butonu sadece iç kısımlarda kullanılmaya uygundur. Sıva altı veya üstü montajlanabilir. Yönsüz bağlantı ve ilk durumuna döndürülebilme (resetlenebilme) özelliğine sahiptir. Yangın ihbar butonu yerden 1,2 m ile 1,6 m yüksekliğe ve iki ihbar butonu arasındaki mesafe maksimum 30 m olacak şekilde monte edilir.

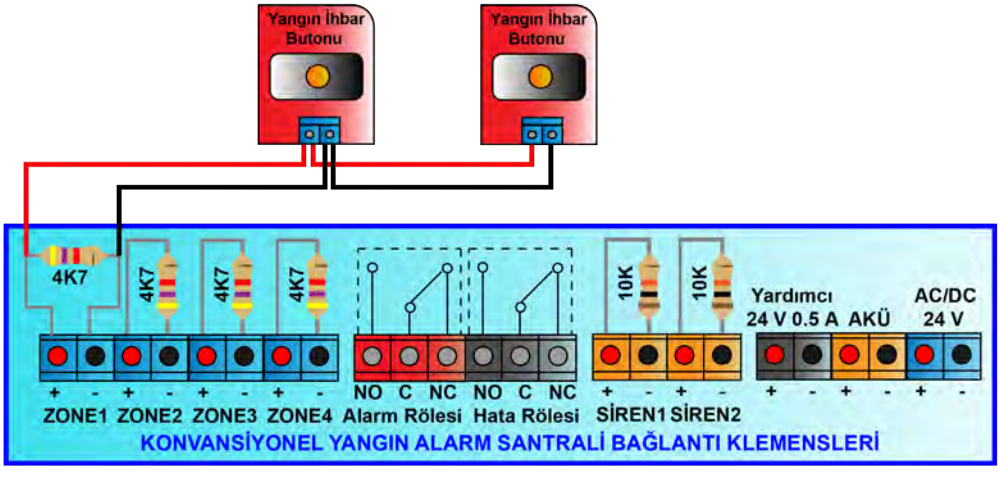
Yangın ihbar butonu konvansiyonel yangın alarm sisteminde ilgili **zone** bağlantısına paralel olacak şekilde bağlanır. Kablo bağlantıları arasında **T** ek yapmak istenmeyen durum olduğu için dedektör tabanına montaj yapılır. Yani en yakın dedektör tabanının **+IN** veya **+OUT** ile **–COM** ucu arasına bağlantı yapılır.

Yangın ihbar butonunda dikkat edilecek nokta, butona basıldığında bağlantı uçları arasında 470 Ω veya 680 Ω gibi bir direnç değeri görünmelidir. Eğer bu direnç değeri yok ise **zone**nun + ve – uçları arasına paralel bağlanacağı için yangın alarm santrali sinyali yangın sinyali olarak değil, kısa devre yani hata sinyali olarak algılar. O zaman da yangın ihbar butonu yangın uyarısı yapmak için değil hata uyarısı yapmak için kullanılmış olur. Buna benzer durumlar ile karşılaşmamak için yangın ihbar butonun bağlantı uçları ölçülür. Butona basılmadığında açık devre, butona basıldığında kısa devre ise butonun bağlantı uçlarının birine seri 470 Ω direnç bağlanarak **zone** hattına bağlantısı yapılır (Görsel 4.61).

Görsel 4.62'de konvansiyonel sistem yangın ihbar butonu bağlantısı görülmektedir.



Görsel 4.61: Konvansiyonel sistemde kullanılan yangın ihbar butonunun direncini ölçme



Görsel 4.62: Konvansiyonel sistem yangın ihbar butonu bağlantısı

İşlem Basamakları

1. Öğretmeninizin kontrolünde Görsel 4.62'deki bağlantıları yaparak Tablo 4.5'teki ilgili yerleri doldurunuz.
2. İş güvenliği ve sağlığı kurallarına uyarak konvansiyonel yangın alarm santralinin kapağını açınız. **Zone** uçlarında hat sonu direnci (4,7 kΩ, 6,8 kΩ) olduğunu, siren uçlarında 10 kΩ hat sonu direnci olduğunu gözlemleyiniz. Eksik ise tamamlayınız.
3. Konvansiyonel yangın alarm santraline bağlantısını yapmak üzere plançeteye iki adet yangın ihbar butonunun taban ayağının montajını yapınız.
4. Görsel 4.62'deki gibi **ZONE1**'e bağlantısını yapınız. Dedektör bağlantısı olmadığı için **ZONE1**'deki 4,7 kΩ-6,8 kΩ hat sonu direncini sökmeyiniz.
5. Yaptığınız yangın ihbar butonunun bağlantısını Görsel 4.62'ye göre kontrol ederek yangın ihbar butonlarının gövdelerini taban ayaklarına oturtunuz.
6. Yangın ihbar butonunun basılı olmadığından emin olunuz. Basılı durumda ise basılı olmayan pozisyona getiriniz.
7. Akü bağlantısını kontrol ederek santrale enerji veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
8. Yangın ihbar butonlarından birincisine basınız. Yangın alarm santralinin yangın sinyali verdiğini gözlemleyiniz.
9. **BuzzerOFF** butonuna basarak **buzzer**ın sustuğunu teyit ediniz.
10. Basılan yangın ihbar butonunun basılı durumda iken basılı olmayan pozisyona getiriniz. Santral üzerindeki **RESET** butonuna basarak yangın alarm santralini yeniden başlatınız.
11. Sistemin normal çalışma moduna döndüğünü gözlemleyiniz.
12. 7, 8, 9 ve 10. işlem basamaklarını 2. yangın ihbar butonu için tekrarlayınız.
13. 1. yangın ihbar butonunun bağlantılarından birini sökünüz. Durumu gözlemleyip not ediniz.
14. 1. yangın ihbar butonunun bağlantısını eski hâline getiriniz. Bağlantıyı Görsel 4.62'ye göre kontrol ediniz.
15. Santral üzerindeki **RESET** butonuna basarak yangın alarm santralini yeniden başlatınız.
16. Görsel 4.62'deki devre şemasını 3 yangın ihbar butonu bağlanacak şekilde organize ederek, plançete üzerine eklentileri yapıp işlem basamaklarını 3. yangın ihbar butonu için tekrarlayınız.
17. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Tablo 4.5: Yangın İhbar Butonunun Ölçüm Sonuçları

Ölçüm Yapılacak Yer	Buton Durumu	Ölçüm Sonucu (R,Ω)
Yangın İhbar Butonunun Bağlantı Klemensleri	Butona Basılı Değil	
	Butona Basılı	

Sorular

1. 3. işlem basamağında neden **zone** üzerindeki hat sonu direnci sökülmeden yangın ihbar butonları bağlanmıştır? Açıklayınız.
2. 12. işlem basamağının sonucunda gözlemlediğiniz durumu sebepleriyle açıklayınız.
3. Görsel 4.62'deki bağlantı şemasına dışardan **push buton** bağlantısı nasıl yapılır? Açıklayarak yeni bağlantı şemasını yani şemanın 3. yangın ihbar butonu ve **push buton** eklenmiş hâlini çiziniz.

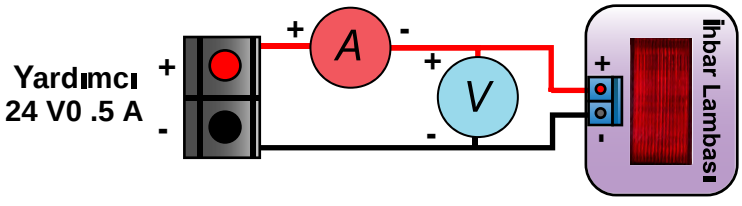
Uygulama Adı: Konvansiyonel Yangın Alarm Santralinde Dedektörlere Paralel İhbar Lambasının Bağlantısı — No.: 6

Amaç: Konvansiyonel yangın alarm santralinde dedektörlere paralel ihbar lambası bağlamak.

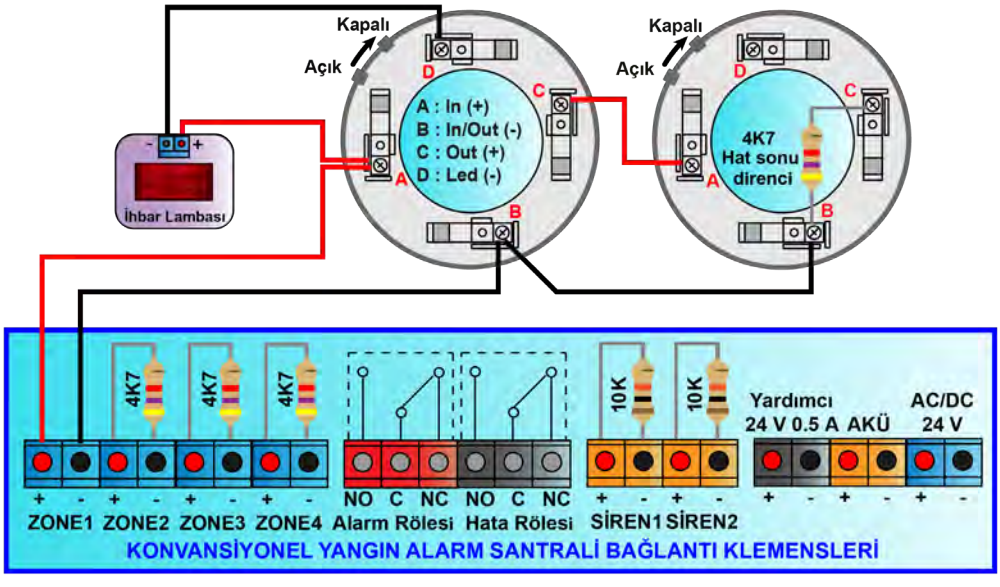
Yangın algılama dedektörlerinin; yüksek tavan, asma tavan, yükseltilmiş taban, girilmesi sakıncalı odalar vb. yerlere montajından dolayı alarm LED'lerinin görülemediği durumlarda, ek bildirim LED'i olarak paralel ihbar lambaları kullanılır.

Paralel ihbar lambasının ilgili dedektörün taban ayağının **+IN** ile **LED-** uçlarına bağlantısı yapılır. Dedektörün taban ayaklarındaki LED- ucu dışardan bir ürünü tetiklemek için – sinyal gönderir.

Görsel 4.63'te paralel ihbar lambası akım ve gerilim değeri ölçme bağlantısı, Görsel 4.64'te ise konvansiyonel sisteme paralel ihbar lambası bağlantısı görülmektedir.



Görsel 4.63: Paralel ihbar lambası akım ve gerilim değerini ölçme bağlantısı



Görsel 4.64: Konvansiyonel sisteme paralel ihbar lambası bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak konvansiyonel yangın alarm santralinin kapağını açınız. **Zone** uçlarında hat sonu direnci (4,7 kΩ, 6,8 kΩ) olduğunu, siren uçlarında 10 kΩ hat sonu direnci olduğunu gözlemleyiniz. Eksik ise tamamlayınız.
2. Görsel 4.63'teki devre şemasının bağlantısını yangın alarm santrali üzerinde bulunan 24 V 0,5 A yardımcı klemens çıkışına yapınız.
3. Akü bağlantısını kontrol ederek santrale enerji veriniz ve **RESET** tuşuna basarak sistemi

resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.

4. Görsel 4.63'teki devre şemasındaki ölçüm sonuçlarını Tablo 4.6'ya yazınız.
5. Sistemin akü bağlantılarını söküp şebeke gerilimini kesiniz.
6. Konvansiyonel yangın alarm santraline Görsel 4.64'teki şemanın bağlantısını yapmak üzere plançeteye 2 adet dedektör taban ayağının ve paralel ihbar lambasının ayağının montajını yapınız.
7. Duman veya ısı dedektörünün gövdesinin altında yazan numaraları takip ederek taban ayağının **ZONE1**'e bağlı olan hat sonu direncini söküp bağlantısını yapınız.
8. 2. dedektör bağlantısını yapınız ve hat sonu direncini takınız.
9. Paralel ihbar lambasının 1. dedektörün taban ayağına bağlantısını yapınız.
10. Yaptığınız dedektör bağlantısını Görsel 4.64'e ve dedektör gövdesinin altına göre kontrol ederek dedektör gövdelerini taban ayaklarına oturtunuz.
11. Paralel ihbar lambasının bağlantı uçlarının yönlerini ve bağlantı klemensinin üzerindeki yazıları kontrol ederek ihbar lambasının kapağını kapatınız.
12. Akü bağlantısını kontrol ederek santrale enerji veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
13. 1. dedektöre duman unsurunu uygulayınız. Yangın alarm santralinin yangın sinyali verdiğini ve paralel ihbar lambasının yandığını gözlemleyiniz.
14. **BuzzerOFF** butonuna basarak **buzzer**ın sustuğunu teyit ediniz.
15. Dedektörün algıladığı duman unsuru dedektörden uzaklaştırarak santral üzerindeki **RESET** butonuna basarak yangın alarm santralini yeniden başlatınız.
16. Sistemin normal çalışma moduna döndüğünü ve paralel ihbar lambasının söndüğünü gözlemleyiniz.
17. 2. dedektöre ısı unsuru uygulayınız. Yangın alarm santralinin yangın sinyali verdiğini ve paralel ihbar lambasının sönük kaldığını gözlemleyiniz.
18. **BuzzerOFF** butonuna basarak **buzzer**ın sustuğunu teyit ediniz.
19. Dedektörün algıladığı ısı unsuru dedektörden uzaklaştırarak santral üzerindeki **RESET** butonuna basıp yangın alarm santralini yeniden başlatınız.
20. Sistemin normal çalışma moduna döndüğünü ve paralel ihbar lambasının sönük kaldığını gözlemleyiniz.
21. Sistemin akü bağlantılarını söküp şebeke gerilimini kesiniz.
22. Görsel 4.64'teki devre şemasını 2 dedektöre de paralel ihbar lambası bağlanacak şekilde organize ederek plançete üzerine eklentileri yapıp işlem basamaklarını tekrarlayınız.
23. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Tablo 4.6: Paralel İhbar Lambasının Ölçüm Sonuçları

Ölçüm Yapılacak Eleman	Gerilim (V)	Akım (A)
Paralel İhbar Lambası		

Sorular

1. Paralel ihbar lambalarını araştırarak ışıklı ve sesli olacak şekilde tasarlayınız.
2. Dedektör bağlantısındaki LED (-) tetik ucunu ve anahtarlama elemanı olarak röle kullanarak paralel ihbar lambası gibi AC 220 V ile çalışan paralel ihbar lambası tasarımının devre şemasını Görsel 4.64'ten hareketle yapınız.

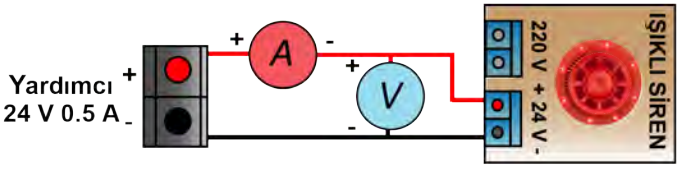
Uygulama Adı: Konvansiyonel Yangın Alarm Santraline Siren Bağlanması — No.: 7

Amaç: Konvansiyonel yangın alarm santraline siren bağlamak.

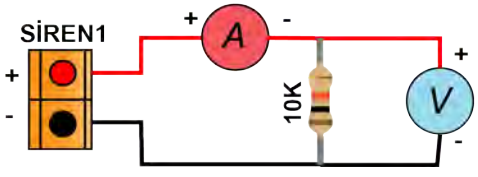
Siren, yangın alarm santralinden gelen tetikleme sinyaline göre sesli ve ışıklı olarak yangın alarm bilgisi vererek insanları uyarır. Sirenlerin flaşörlü (ışıklı) ve flaşörsüz (ışıksız) çeşitleri vardır.

Görsel 4.65'te siren elemanının akım ve gerilim değerini ölçme, Görsel 4.66'da siren çıkışının akım ve gerilim değerini ölçme bağlantıları görülmektedir.

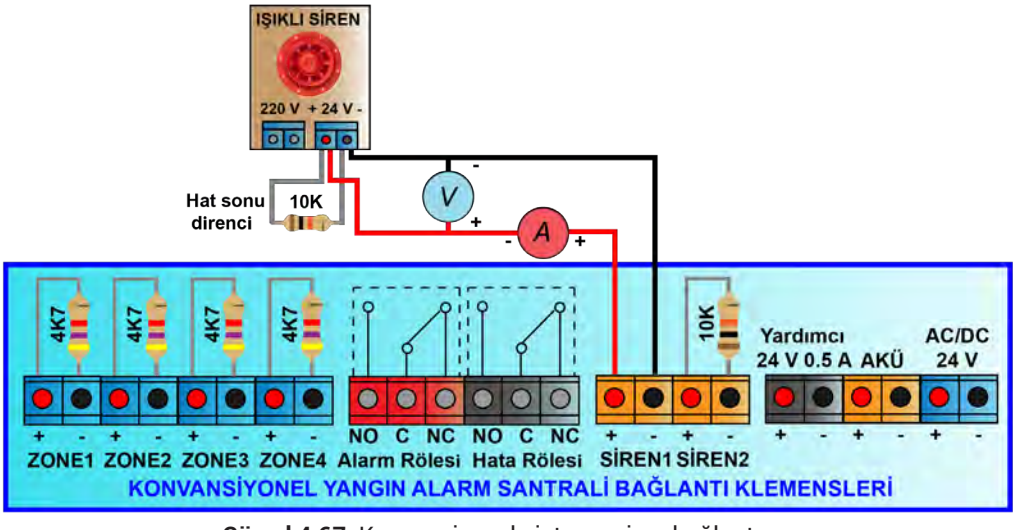
Görsel 4.67'de ise konvansiyonel siren bağlantısı görülmektedir.



Görsel 4.65: Siren elemanının akım ve gerilim değerini ölçme bağlantısı



Görsel 4.66: Siren çıkışının akım ve gerilim değerini ölçme bağlantısı



Görsel 4.67: Konvansiyonel sisteme siren bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak konvansiyonel yangın alarm santralinin kapağını açı nız. **Zone** uçlarında hat sonu direnci (4,7 kΩ, 6,8 kΩ) olduğunu, siren uçlarında 10 kΩ hat sonu direnci olduğunu gözlemleyiniz. Eksik ise tamamlayınız. 3. paralel ihbar lambasını araştırma yaparak ışıklı ve sesli olacak şekilde tasarlayınız.

2. Görsel 4.65'teki devre şemasının bağlantısını yangın alarm santrali üzerinde bulunan 24 V 0,5 A yardımcı klemens çıkışına yapınız.
3. Akü bağlantısını kontrol ederek santrale enerji veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
4. Görsel 4.65'teki devre şemasındaki ölçüm sonuçlarını Tablo 4.7'ye yazınız.
5. Sistemin akü bağlantılarını söküp şebeke gerilimini kesiniz.
6. Görsel 4.66'daki devre şemasının bağlantısını yangın alarm santrali üzerinde bulunan istediğiniz siren çıkışına yapınız.
7. Akü bağlantısını kontrol ederek santrale enerji veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
8. Görsel 4.66'daki devre şemasında, yangın alarm santralini çalışma modlarına kendiniz getirerek ölçüm sonuçlarını Tablo 4.8'e yazınız.
9. Sistemin akü bağlantılarını söküp şebeke gerilimini kesiniz.
10. Görsel 4.67'deki devre şemasının bağlantısını yangın alarm santrali üzerinde bulunan istediğiniz siren çıkışına yapınız.
11. Akü bağlantısını kontrol ederek santrale enerji veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
12. Görsel 4.67'deki devre şemasında, yangın alarm santralini çalışma modlarına kendiniz getirerek ölçüm sonuçlarını Tablo 4.8'e yazınız.
13. 10, 11, 12 ve 13. işlem basamaklarına göre Görsel 4.67'deki devre şemasına 2. sireni de bağlayarak Tablo 4.8'deki ilgili yerleri doldurunuz.
14. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Tablo 4.7: Sirenin Ölçüm Sonuçları

Ölçüm Yapılacak Eleman	Gerilim (V)	Akım (A)
Siren		

Tablo 4.8: Yangın Alarm Santralinin Çalışma Modlarındaki Siren Gerilim ve Akım Ölçüm Sonuçları

Alarm Santralinin Çalışma Modu	SİREN No.	Görsel No	SİREN Gerilimi (V)	SİREN Akımı (A)	Santralin Durumu İstenen Çalışma Modunda
DENGE Modu		Görsel 4.66			Çalışıyor / Çalışmıyor
ALARM Modu					Çalışıyor / Çalışmıyor
HATA Modu					Çalışıyor / Çalışmıyor
DENGE Modu		Görsel 4.67			Çalışıyor / Çalışmıyor
ALARM Modu					Çalışıyor / Çalışmıyor
HATA Modu					Çalışıyor / Çalışmıyor
DENGE Modu		Görsel 4.67 2 Sirenli Bağlantı Ölçümleri			Çalışıyor / Çalışmıyor
ALARM Modu					Çalışıyor / Çalışmıyor
HATA Modu					Çalışıyor / Çalışmıyor

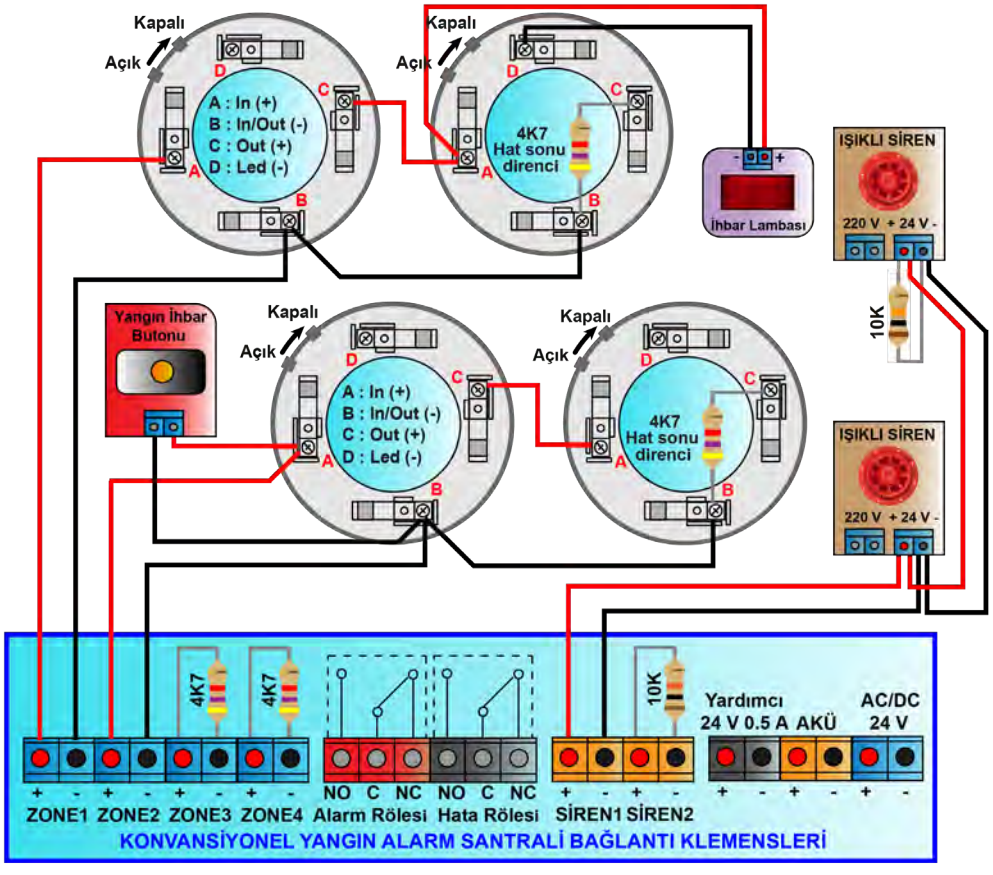
Soru

1. Sireni ışıklı ve sesli olarak araştırıp tasarlayınız.

Uygulama Adı: Konvansiyonel Yangın Alarm Santralinde 2 Zone Kullanılarak Dedektör, Yangın İhbar Butonu, Paralel İhbar Lambası ve Siren Bağlanması **No.: 8**

Amaç: Konvansiyonel yangın alarm santralinde 2 zone kullanarak dedektör, yangın ihbar butonu, paralel ihbar lambası ve siren bağlamak.

Görsel 4.68'de 2 **zone**lu bağlantı şeması verilmiştir. Her **zone** için bağlantı ve montaj işlemlerini ayrı ayrı olacak şekilde yapmanız önerilir. Sistemi devreye almadan önce test etmeniz işinizi kolaylaştıracaktır.



Görsel 4.68: Konvansiyonel yangın alarm santraline 2 zone bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak konvansiyonel yangın alarm santralinin kapağını açınız. **Zone** uçlarında hat sonu direnci (4,7 kΩ, 6,8 kΩ) olduğunu, siren uçlarında 10 kΩ hat sonu direnci olduğunu gözlemleyiniz. Eksik ise tamamlayınız.
2. Görsel 4.68'deki devre şemasının bağlantısını yapınız.
3. Akü bağlantısını kontrol ederek santrale enerji veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
4. Kurduğunuz iki **zone**lu sistemde kullanılan elemanları test ederek sistemi devreye alınız.
5. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Soru

1. Sistemi nasıl test edip devreye aldığınızı açıklayınız.

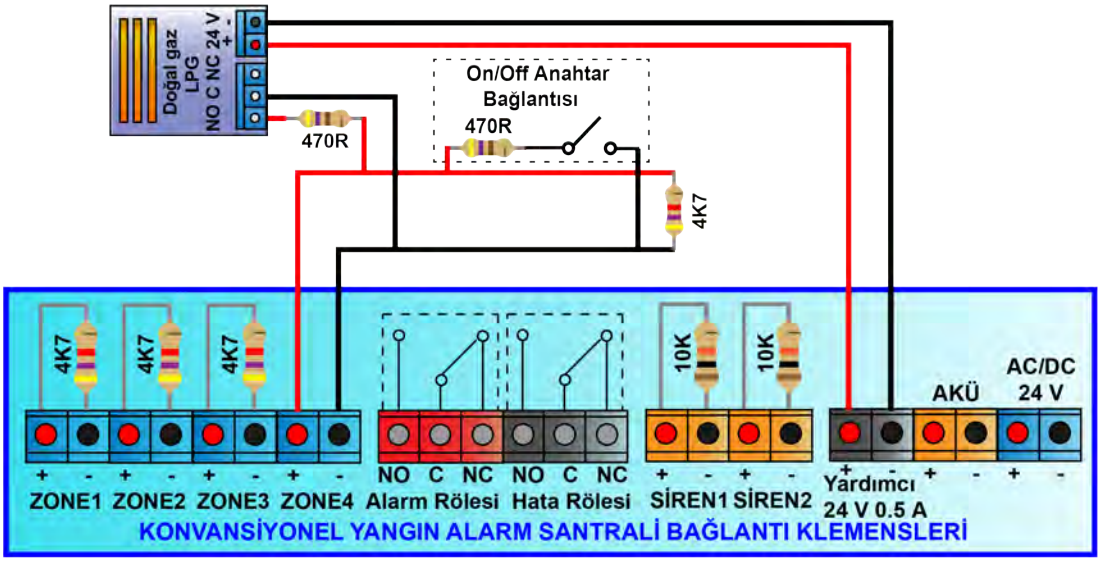
Uygulama Adı: Konvansiyonel Yangın Alarm Santraline Doğal Gaz-LPG Dedektörü Bağlanması — No.: 9

Amaç: Konvansiyonel yangın alarm santraline doğal gaz-LPG dedektörü bağlamak.

Doğal gaz veya LPG dedektörleri, herhangi bir sebepten dolayı gaz kaçağı olması ve ortamdaki gaz seviyesinin belirlenen seviyeyi aşması durumunda alarm vermektedir.

Gaz dedektörlerini diğer dedektörlerden ayıran özellik ise dört kablo ile bağlanması ve çıkışlarında röle kontağının bulunmasıdır. Röle kontakları kuru kontak anahtarlaması yapmaktadır. Çıkışında kuru kontak anahtarlaması yapan dedektör direkt kullanılmaz. Çünkü **zone** hattını kısa devre yapabilir. Bunun için genellikle **zone** hattına ön direnç (yaklaşık 470 Ω) ile birlikte bağlanır. Bu ön direncin değeri ise marka ve modele göre farklılık gösterebilir. Bunun için mutlaka yangın alarm santralinin katalog bilgilerine bakılmalıdır.

Görsel 4.69'da **zone** hattına hem doğal gaz-LPG dedektörünün bağlantısı hem de **ON-OFF** anahtar bağlantısı görülmektedir. Çıkışında kuru kontak anahtarlaması yapan elemanlar mutlaka **zone** hattına ön direnç ile birlikte bağlanır.



Görsel 4.69: Konvansiyonel sisteme doğal gaz-LPG dedektörü bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak Görsel 4.69'daki bağlantı şemasını kullanacağınız elemanları plançeteye montajını yaparak oluşturunuz.
2. Öğretmeninizin kontrolünde akü bağlantısını kontrol ederek santrale şebeke gerilimini veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
3. Doğal gaz-LPG dedektörüne algıladığı unsuru (gaz) uygulayınız. Yangın alarm santralinin yangın sinyali verdiğini gözlemleyiniz.
4. **BuzzerOFF** butonuna basarak **buzzer**ın sustuğunu teyit ediniz.
5. Dedektörün algıladığı unsuru dedektörden uzaklaştırarak santral üzerindeki **RESET** butonuna basıp yangın alarm santralini yeniden başlatınız.

6. Sistemin normal çalışma moduna döndüğünü gözlemleyiniz.
7. **ON-OFF** anahtarına basınız. Yangın alarm santralinin yangın sinyali verdiğini gözlemleyiniz. 4, 5 ve 6 No.lu işlem basamaklarını tekrar uygulayınız.
8. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Sorular

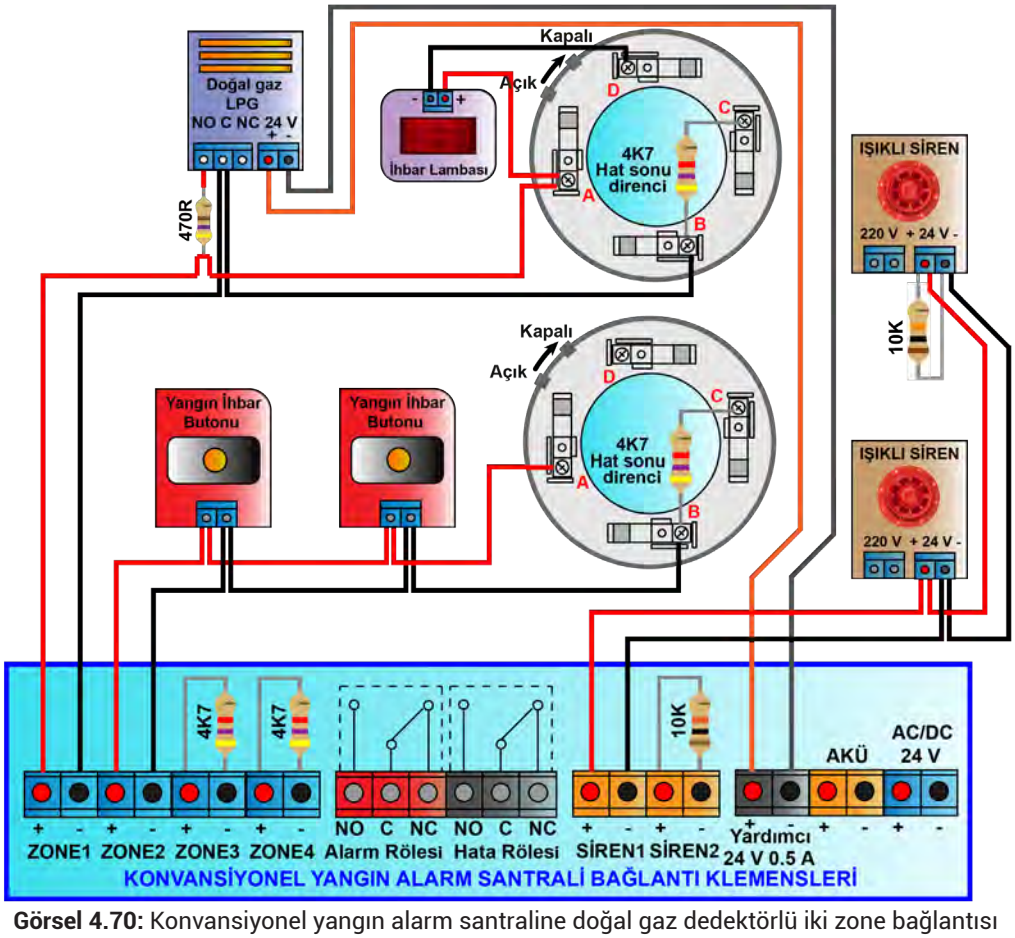
1. Kuru kontak anahtarlaması nedir? Açıklayınız.
2. Çıkışında kuru kontak anahtarlaması yapılan diğer ürünleri araştırınız. Bu ürünlerin yangın alarm santrallerinde ne amaç ile kullanılabileceğini tartışınız.

NOT DEFTERİ

Uygulama Adı: Konvansiyonel Yangın Alarm Santraline Bütün Ürünlerin Bağlantısının Yapılması — No.: 10

Amaç: **Konvansiyonel yangın alarm santraline bütün ürünlerin bağlantısını yapmak.**

Görsel 4.70'te 2 **zone**lu bağlantı şeması verilmiştir. Her **zone** için bağlantı ve montaj işlemlerini ayrı ayrı yapmanız önerilir. Sistemi devreye almadan önce test etmeniz işinizi kolaylaştıracaktır.



Görsel 4.70: Konvansiyonel yangın alarm santraline doğal gaz dedektörlü iki zone bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak konvansiyonel yangın alarm santralinin kapağını açı nız. **Zone** uçlarında hat sonu direnci (4,7 kΩ, 6,8 kΩ) olduğunu, siren uçlarında 10 kΩ hat sonu direnci olduğunu gözlemleyiniz. Eksik ise tamamlayınız.
2. Görsel 4.70'teki devre şemasının bağlantısını yapınız.
3. Öğretmeninizin kontrolünde akü bağlantısını kontrol ederek santrale enerji veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
4. Kurduğunuz iki **zone**lu sistemde kullanılan elemanları test ederek sistemi devreye alınız.
5. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Soru

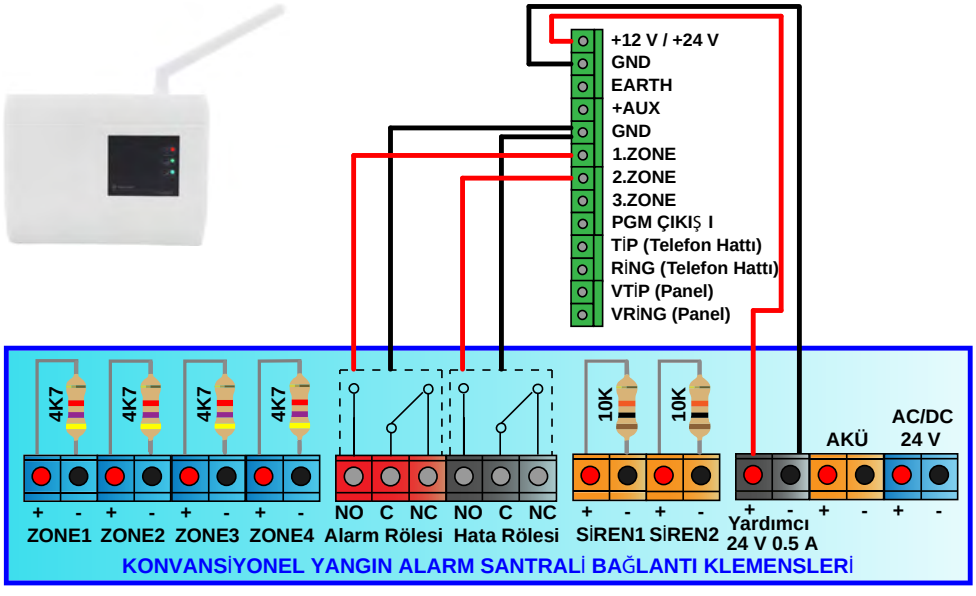
1. Sistemi nasıl test edip devreye aldığınızı açıklayınız.

Uygulama Adı: Konvansiyonel Yangın Alarm Santralinde GSM Modülü Kullanımı — No.: 11

Amaç: Konvansiyonel yangın alarm santralinde GSM modülü kullanmak.

GSM modülü ile birlikte gelen program aracılığı ile **ZONE1**, **ZONE2** ve **ZONE3** sekmelerine ilgili numaralara gönderilecek SMS bilgilerini oluşturunuz. Hatta her bir **zone**a arama yaptığında söyleyeceği ses kaydı bilgilerini yükleyiniz. Örnek olarak **ZONE1**'e "yangın alarmı"**, ZONE2**'ye de "sistem hatası" bilgilerini yükleyebilirsiniz.

Görsel 4.71'deki devrede GSM modülünün **zone** kuru kontak algılama girişleri, yangın alarm santralinin alarm ve hata rölelerine bağlanmıştır.



Görsel 4.71: Konvansiyonel yangın alarm santraline GSM modülü bağlantısı

İşlem Basamakları

1. GSM modülünün ZONE1 ve ZONE2 girişlerine göndereceği mesaj veya ses kaydını giriniz.
2. İş güvenliği ve sağlığı kurallarına uyarak Görsel 4.71'deki bağlantı şemasını kullanacağınız elemanların plançeteye montajını yapınız.
3. Öğretmeninizin kontrolünde akü bağlantısını kontrol ederek santrale şebeke gerilimini veriniz ve RESET tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
4. Yangın alarm santralini hatalı çalışma moduna getiriniz.
5. GSM modülünde 2 No.lu **zone**a yüklenen SMS'in veya telefon çağrısının gelmesini bekleyiniz.
6. Yangın alarm santralini hatalı çalışma modundan alarm moduna getiriniz.
7. GSM modülünde1 No.lu **zone**a yüklenen SMS'in veya telefon çağrısının gelmesini bekleyiniz.
8. Sistemi normal çalışma moduna getiriniz.
9. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

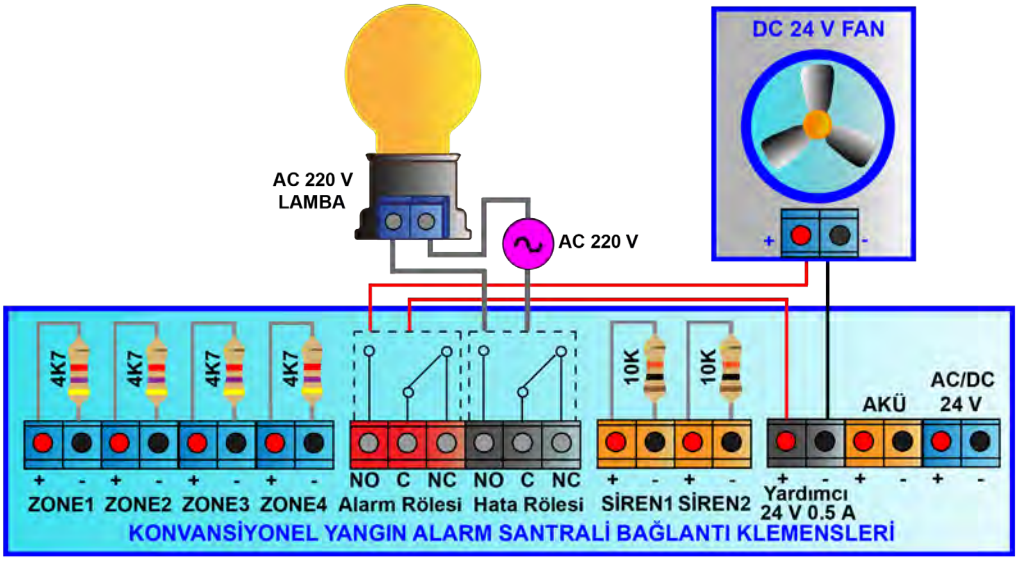
Soru

1. GSM modülünün besleme gerilimi 12 V ise yangın alarm panelinden besleme gerilimini nasıl elde edersiniz? Araştırarak açıklayınız.

Uygulama Adı: Konvansiyonel Yangın Alarm Santraline DC 24 V Fan Ve AC 220 V Lamba Bağlantısı Yapılması — No.: 12

Amaç: Konvansiyonel yangın alarm santraline DC 24 V fan ve AC 220 V lamba bağlantısı yapmak.

Görsel 4.72'deki bağlantı şemasında hata anında yangın alarm santralinin bulunduğu odanın ışığının (AC 220 V lamba) yanması, aynı zamanda da yangın anında ortamda bulunan dumanın havalandırma sistemi ile dışarı atılması sağlanmaktadır. Yangın alarm santralinin hata ve alarm röleleri üzerinden DC ve AC sinyal anahtarlaması gösterilmiştir.



Görsel 4.72: Konvansiyonel yangın alarm santraline 24V fan ve AC 220V lamba bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak Görsel 4.72'deki bağlantı şemasını kullanacağınız elemanların plançeteye montajını yapınız.
2. Öğretmeninizin kontrolünde akü bağlantısını kontrol ederek santrale şebeke gerilimini veriniz ve **RESET** tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
3. Yangın alarm santralini hatalı çalışma moduna getiriniz. AC 220 V lambanın yandığını gözlemleyiniz.
4. Yangın alarm santralini denge çalışma moduna alarak lambanın söndüğünü gözlemleyiniz.
5. Yangın alarm santralini denge çalışma modundan alarm moduna getiriniz. DC 24 V fanın çalıştığını gözlemleyiniz.
6. Santrali resetleyerek sistemi denge çalışma moduna getiriniz ve fanın söndüğünü gözlemleyiniz.
8. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

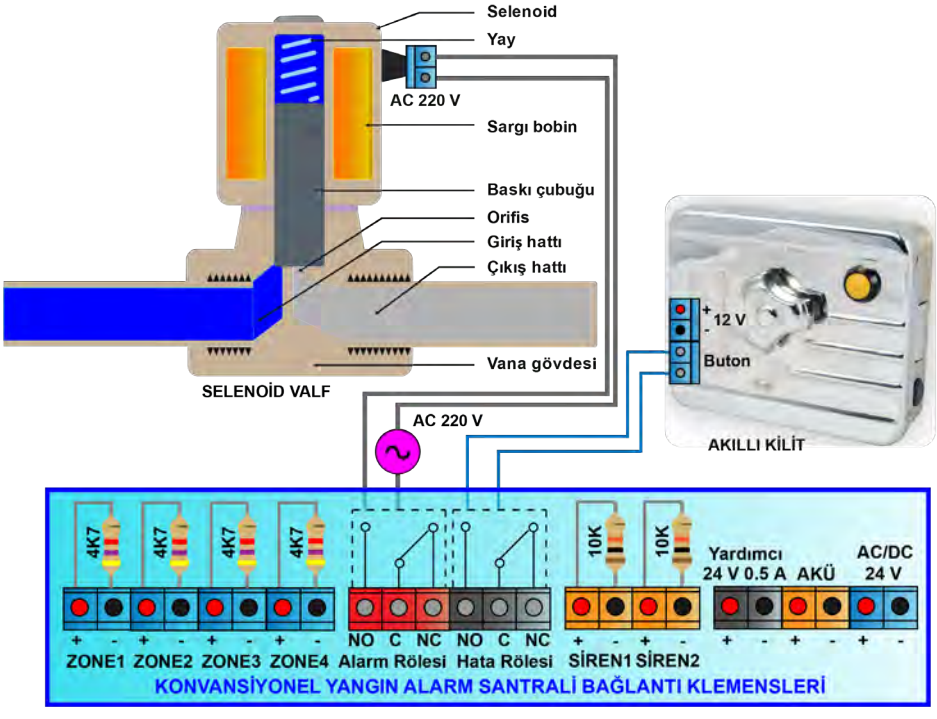
Sorular

1. DC 24 V fan yerine 2 tane DC 12 V fan bağlantısının nasıl yapılacağını açıklayınız.
2. DC 24 V fan yerine 1 tane DC 12 V fan bağlantısının nasıl yapılacağını açıklayınız
3. DC 24 V fan yerine AC 220 V fan bağlantısının nasıl yapılacağını açıklayınız.

Uygulama Adı: Konvansiyonel Yangın Alarm Santraline Selenoid Valf ve Akıllı Kilit Bağlantısı Yapılması — No.: 13

Amaç: Konvansiyonel yangın alarm santraline selenoid valf ve akıllı kilit bağlantısı yapmak.

Görsel 4.73'teki bağlantı şemasında hata anında yangın alarm santralinin bulunduğu odanın kapısındaki akıllı kilidin kuru kontak anahtarlaması ile açılması, aynı zamanda da yangın anında ortamda bulunan gazın selenoid valf ile kesilmesi sağlanmaktadır. Yangın alarm santralinin hata ve alarm röleleri üzerinden kuru kontak ve AC sinyal anahtarlaması gösterilmektedir. Selenoid gaz valfleri genellikle ortamın doğal gaz girişine bağlanır.



Görsel 4.73: Konvansiyonel yangın alarm santraline AC 220 V selenoid valf ve akıllı kilit bağlantısı

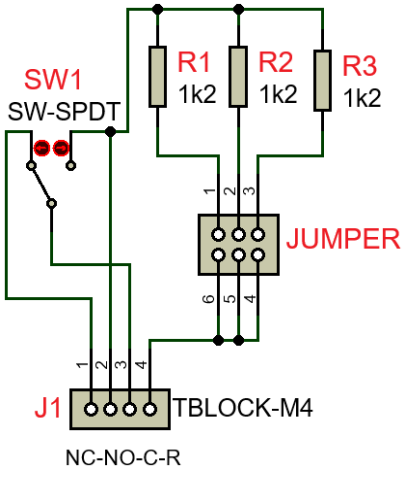
İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak Görsel 4.73'teki bağlantı şemasını kullanacağınız elemanların plançeteye montajını yapınız.
2. Görsel 4.73'teki akıllı kilidin besleme gerilimi olan 12 V bağlantısını yapınız.
3. Öğretmeninizin kontrolünde akü bağlantısını kontrol ederek santrale şebeke gerilimini veriniz ve RESET tuşuna basarak sistemi resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
4. Yangın alarm santralini hatalı çalışma moduna getiriniz. Akıllı kilidin, çalıştığını (kilit dilinin içeri çekildiğini) gözlemleyiniz.
5. Yangın alarm santralini denge çalışma moduna alarak akıllı kilidin kapandığını gözlemleyiniz.
6. Yangın alarm santralini denge çalışma modundan alarm moduna getiriniz. AC 220 V selenoid valfin gaz geçiş kanalını kapattığını gözlemleyiniz.
7. Santrali resetleyerek sistemi denge çalışma moduna getiriniz ve selenoid valfin gaz geçiş kanalını açtığını gözlemleyiniz.
8. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Uygulama Adı: Yangın İhbar Butonu Yapımı — No.: 14

Amaç: **Yangın ihbar butonu yapmak.**

Görsel 4.74'te yangın ihbar butonunun devre şeması görülmektedir. Devre şemasının çıkış uçlarında dörtlü NO:0 klemens mevcuttur. Bu klemensin C-R uçları arası butona basıldığında alarm direncini vermektedir. Alarm direncini üçlü **jumper**ları birleştirerek sisteminize uygun olacak şekilde ayarlayınız. NO ve NC uçları ise C ucu ile birlikte röle uçları gibi kapalı-açık kuru kontak uçlarıdır.



Görsel 4.74: Yangın ihbar butonu devre şeması

İşlem Basamakları

1. Görsel 4.74'teki devrenin baskı devre şemasını baskı devre programında çiziniz.
2. Yaptığınız çizimin çıktısını alıp iş güvenliği ve sağlığı önlemlerini alarak bakır plakete aktarıp çıkarınız ve deliklerini deliniz.
3. Devre elemanlarının sağlamlık kontrollerini yapıp bakır plaket üzerine montajını yaparak elektronik kartı üretiniz.
4. Üretimini yapmış olduğunuz yangın ihbar butonu devresini öğretmeninizin gözetiminde yangın alarm santraline bağlayarak çalışmasını deneyiniz.

Sorular

1. SW anahtarı ve jamper'ın görevi nedir? Açıklayınız.
2. Devresini yaptığınız yangın ihbar butonunda SW anahtarı ve jumper hangi konumda iken atölyedeki yangın alarm santralinde kullanabilirsiniz? Açıklayınız.
3. Atölyenizdeki yangın alarm santralinde kullanılan yangın ihbar butonunun açık şemasını çiziniz.

NOT DEFTERİ

5.
ÖĞRENME
BİRİMİ



HIRSIZ ALARM SİSTEMLERİ

NELER ÖĞRENECEKSİNİZ ?

- Hırsız Alarm Santrali
- Hırsız Alarm Sistemi Donanımları
- Hırsız Alarm Sisteminin Programlanması

TEMEL KAVRAMLAR

kontrol paneli
PIR dedektörü
hareket
keypad
siren

5. HIRSIZ ALARM SİSTEMLERİ

Başkasının malını çalan kişiye **hırsız**, olaya ise **hırsızlık** denir. Hırsız alarm sistemleri, evlerde veya iş yerlerinde yaşanabilecek hırsızlık, gasp, tehdit, yangın gibi durumlarda mülk sahiplerini ve çevreyi haberdar eden akıllı elektronik sistemlerdir.

Yangın alarm sistemleri, yangın sonucunda ortaya çıkan unsurları algılayıp ihbar cihazları aracılığı ile insanları uyaran sistemlerken hırsız alarm sistemleri ise hırsızlık olayı başlamadan önce ortaya çıkan unsurları algılayıp ihbar cihazları aracılığı ile insanları uyaran sistemlerdir. Hırsız alarm sistemlerinin amacı hırsızlık olayı başlamadan önce hırsızı caydırmak, hırsızlık olayında oluşabilecek can ve mal kayıplarını olabildiği kadar engellemektir.

Hırsız alarm sistemleri; bankalarda, kuyumcularda, evlerde, ofislerde, iş yerlerinde, depolarda, okullarda ve tüm benzer kapalı alanlarda yaygın olarak kullanılmaktadır. Görsel 5.1'de hırsız alarm sistemlerinde kullanılan farklı ürünler görülmektedir.



Görsel 5.1: Hırsız alarm sistemlerinde kullanılan farklı ürünler

Hırsız Alarm Sistemlerinde Kullanılan Ürünler

Hırsız alarm sistemlerinde kullanılan ürünlerin hemen hemen hepsinin kablolu ve kablosuz (**RF, Wi-Fi**) modelleri vardır.

1. Hırsız Alarm Santrali: Hırsız alarm sisteminin kontrol merkezidir. Dedektörlerden (PIR, cam kırılma, beam vb.) veya manyetik kontaklardan alınan elektriksel sinyalleri, yapılan programa göre yorumlayarak çıkışlarına bağlanan siren, GSM modülü, röle, ikaz LED'leri gibi elemanların çalıştırılmasını sağlar.

2. Dedektörler: Hareket, ses ve ışın gibi hırsız unsurlarını sensörleri vasıtasıyla algılayarak elektriksel sinyal olarak hırsız alarm santraline aktaran elemanlardır. Dedektörlerden gelen bilgiye göre hırsız alarm santrali çalışma durumunu belirler.

3. Keypad (Tuş Takımı, Şifre Paneli): Hırsız alarm santraline şifrelerin girildiği, sistemin kurulmasını ve kapanmasını (ON/OFF) sağlayan tuş takımıdır. LED'li veya LCD'li modelleri vardır.

4. Sirenler: Hırsız alarm santrali, dedektörlerden veya manyetik kontaktan "Hırsız var." bilgisini aldıktan sonra flaşörlü veya flaşörsüz sireni aktif eder. Yüksek ses şiddeti ve fasılalı ışık sinyali

ile ortamın tehlikeli olduğu haberini verir. Kablosu kesildiğinde veya yerinden söküldüğünde de alarm durumuna geçer.

5. Manyetik Kontak: Kapının veya pencerenin açılmasını algılayan cihazdır. Açılabilir hareketli noktalara monte edilir.

6. Uzaktan Kumanda: Hırsız alarm santralinin uzaktan kurulup kapanmasını sağlayan cihazdır.

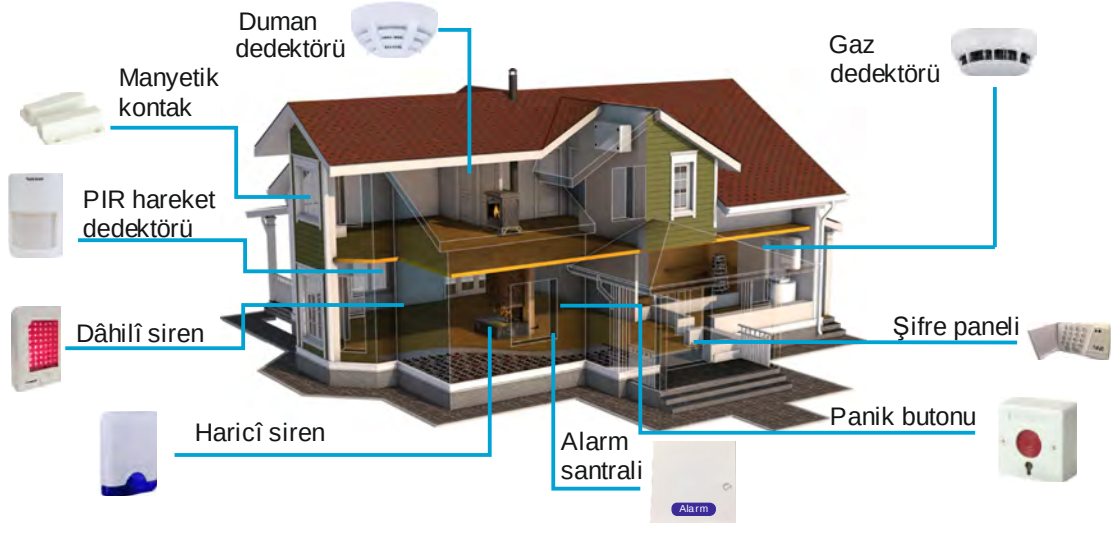
7. Diğer Dedektörler: Yangın alarm sistemlerinde kullanılan duman, ısı, alev, doğal gaz dedektörlerinin çıkışında röle kontrollü olanları veya hırsız alarm santrallerinde yangın **zone** girişi olan dedektörler de hırsız alarm sistemlerinde kullanılabilir.

8. Zone Genişleme Modülleri: Hırsız alarm santrallerinin çıkışındaki **zone** sayısını ihtiyaca göre arttırmak için kullanılır.

9. GSM ve Telefon Arayıcı Modülü: GSM modülleri veya telefon arayıcı modülleri hırsız alarm santralinden gelen uyarı sinyaline göre hafızasında kayıtlı olan telefon numaralarını sıra ile arar ve hafızasında yüklü olan sesli mesajı telefon hattının karşısındaki kişilere iletmek için kullanılır.

10. Haber Alma Merkezleri ve İOS-Android Uygulama Yazılımları: Hırsız alarm santralinin sürekli durumunun izlenmesi için haber alma merkezleri ile iletişim sağlanabilir. Haber alma merkezleri hırsız alarm santrallerinin izlenmesi için yıllık belli bir ücret karşılığında bu hizmeti vermektedir. Son zamanlarda İOS-Android uygulamaların gelişmesi ile hırsız alarm santrali üreticileri, İOS-Android uygulamalarla hırsız alarm santrallerinin programlamasını yapabilmektedir. Hatta hırsız alarm santrali üreticileri, kullanıcılarının İOS-Android uygulama yazılımları ile sistemlerini kendilerinin kontrol edebilmesini ve izleyebilmesini sağlamaktadırlar.

Görsel 5.2'de hırsız alarm sistemlerinde kullanılan ürünlerin örnek bir mimari yapı üzerindeki yerleşimi görülmektedir.



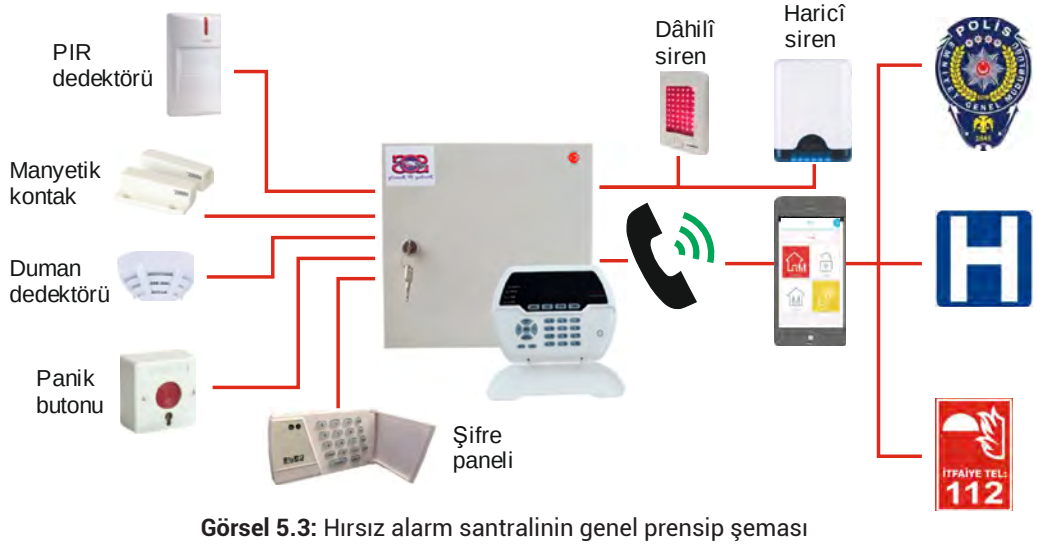
Görsel 5.2: Hırsız alarm sisteminin örnek bir mimari yapı üzerinde yerleşimi

5.1. HIRSIZ ALARM SANTRALİ

Hırsız alarm santrali ile bina içerisinde programlanan saatler arasında kontrol sağlanmaktadır. Hırsız alarm santrali, hırsızlık olayına maruz kalacak fiziksel mekânı olayın başında algılayıp gereken önlemlerin alınması için belli yerlere komut veren, programlanabilir bir kontrol sistemidir.

Eğer hırsız alarm santrali hırsızlık olayının başında algılama yapamaz ise sonucunda can ve mal kaybı gerçekleşebilir.

Hırsız alarm santrallerinin genel çalışma ilkesi, dedektörlerden veya manyetik kontaklardan gelen elektriksel sinyallere göre çıkışında sesli ve ışıklı uyarı cihazlarını, telefon arayıcı ürünlerini veya internet hattından haber alma merkezlerini harekete geçirmesi prensibine dayanır. Son zamanlarda hırsız alarm santralleri için geliştirilen İOS-Android uygulamalar sayesinde cep telefonlarındaki uygulama programlarına da ilgili uyarı mesajları gelebilmektedir. Görsel 5.3'te bir hırsız alarm santralinin genel prensip şeması görülmektedir.



Görsel 5.3: Hırsız alarm santralinin genel prensip şeması

Hırsız Alarm Santrallerinin Çalışma Modları

Hırsız alarm santrallerinin çalışma modları, genel olarak hırsız alarm santrali kuruluyken ya da kurulu değilken incelenir. Hırsız alarm santrali kurulu değilken (bekleme-standby modu) kontrol edilen bölgedeki (**zone**) dedektör ve manyetik kontakların seviyesinde bir değişiklik olsa bile alarm modunda çalışmaz ve bağlı olan sirenler uyarı vermez. Sadece ilgili **zone**a ait uyarılar **keypad**de **zone** numarasına ait olan LED'lerin yanık-sönük durumuna göre izlenebilir. **Keypad** aracılığı ile kullanıcı şifresi girilip hırsız alarm santrali kurulduğunda çalışma modları aşağıdaki gibi isimlendirilir:

• Hırsız alarm santrali **denge modunda:** Kontrol edilen bölgenin (**zone**) dedektör ve manyetik kontaklarının seviyesinde değişme yok demektir. Bu durum hırsız alarm santraline bağlı **keypad**deki bir LED uyarıcı ile belirtilir.

• Hırsız alarm santrali **hata modunda:** Dedektörlerde, manyetik kontaklarda, iletim hatlarında açık devre hatası veya kontrol panelinde bir arıza (açık devre, kısa devre) var demektir. Bu durum **keypad**deki LED'lerin kesik kesik yanması ve kesik kesik sesli uyarıcı sinyaller ile belirtilir.

• Hırsız alarm santrali **alarm modunda:** Kontrol edilen bölgelerin birinde veya birden fazlasında hırsızlık teşebbüsü gerçekleşmiş demektir. Bu durumda hırsız alarm santrali, sesli uyarı verebilmek için sireni, ışıklı uyarı vermek için **keypad**deki kırmızı LED'leri kullanır. Aynı zamanda GSM modülü veya internet ile uzaktan kontrol eden haber alma merkezlerine ve İOS-Android uygulamalara gerekli uyarıları yapar.

NOT Hırsız alarm santrallerinde çalışma modlarının **keypad** üzerindeki belirtileri kullanılan marka ve modele göre farklılık gösterebilir.

Hırsız alarm santralleri, kablolu sistem ve kablosuz sistem olmak üzere iki çeşittir. Kablolu ve kablosuz sistemlerde kullanılan dedektör, manyetik kontak, **keypad**, siren vb. ürünlerin çalışma mantığı aynıdır. Tek fark ürünlerin besleme gerilimleri ve hırsız alarm panelleri ile iletişim yöntemidir. Adından da anlaşılacağı gibi kablolu sistemlerde kullanılan ürünlerin hırsız alarm santrali ile bağlantısı kablo üzerinden yapılır. Kullanılan kablodan ürünlere hırsız alarm santrali üzerinden hem besleme gerilimleri hem de data bilgileri iletilir. Kablosuz hırsız alarm sistemlerinde ise ürünlerin besleme gerilimleri genellikle pil ile data bilgisi ise **RF** veya **Wi-Fi** üzerinden gerçekleştirilir.

Hırsız alarm santralinin montajı, genellikle sistem odalarının içerisine veya sigorta panolarının yanına yapılır. Mümkün olduğunca kolay ulaşılamayacak yüksekliğe montajlanması uygun olacaktır. Hırsız alarm santralinin montajının bulunduğu yerde hırsız alarm santraline erişimi ve müdahaleyi engellemek için mutlaka manyetik kontak veya hareket dedektörü olmalıdır. Hırsız alarm santrali kutusunun kapağında yetkisiz kişiler tarafından açılması durumunda alarm oluşturabilecek bir sabotaj **switch** olmalıdır.

5.1.1. Kablolu Sistem Hırsız Alarm Santrali

Görsel 5.4'te farklı modellerdeki kablolu hırsız alarm santralleri görülmektedir.



Görsel 5.4: Farklı modellerdeki kablolu hırsız alarm santralleri

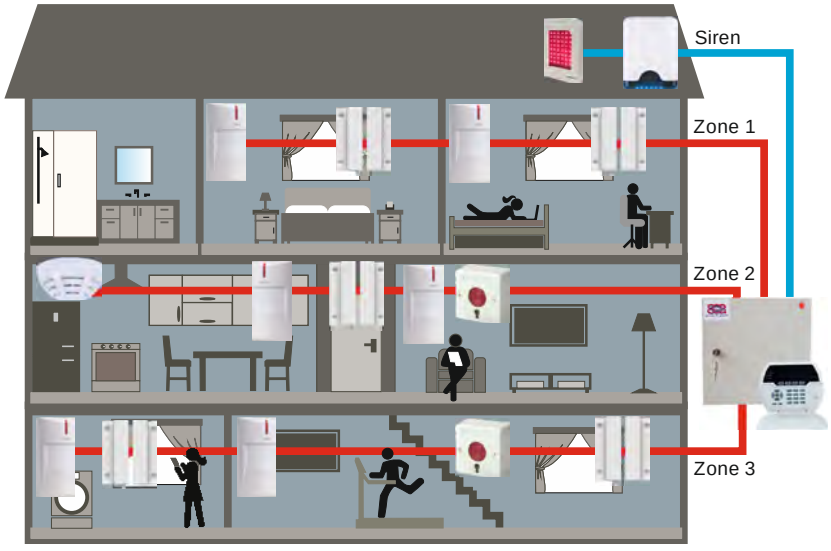
Kablolu hırsız alarm santrali, çıkışında kontrol edeceği **zone** (bölge) sayısına göre isimlendirilir. Her **zone**a genel olarak bir dedektör (PIR, cam kırılma dedektörü, beam vb.) ve manyetik kontak bağlanabilir.

Kablolu hırsız alarm santralinin seçimi montaj yapılacak mimari yapının özelliğine göre olmalıdır. Mimari yapı keşfinde, kullanılacak ürünlerin hepsinin yerleri ve adetleri belirlenir. Kablolu hırsız alarm santrali ile ürünlerini birbirine bağlamak için kabloların nerden, nasıl geçeceği; hangi ürünlerin hangi kablo ile ilgili **zone**a nasıl bağlanacağı belirlenmelidir. Kullanılacak **zone** sayısına ve maliyetine göre kablolu hırsız alarm santrali seçilmelidir.

Kablolu hırsız alarm santrali alınırken yapılan genel hata ise toplam ürün sayısına göre ve her ürün her bir **zone**a ayrı ayrı bağlantı yapılacak gibi düşünülüp o kadar **zone** çıkış sayısına göre seçim yapılmasıdır. Bu da maliyeti arttırmaktadır.

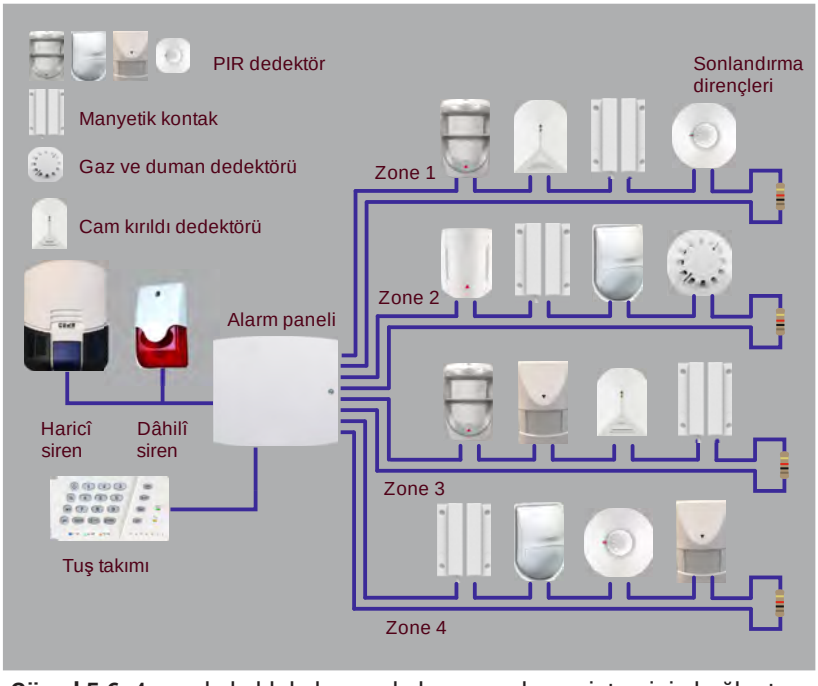
Bir **zone**a bağlanacak dedektör sayısını da yine seçilen bölgenin mimari yapısı belirlemelidir. Örnek olarak 3 katlı bir binanın her katında 2 dedektör varsa 3 katın tamamını bir bölge yapıp hırsız alarm santralinin bir bölgesine bu 3 katı bağlamak son derece yanlış bir uygulamadır. Kullanılan hırsız alarm santralinin **zone** sayısına göre her dedektör veya her kat ayrı bir bölge olarak tanımlanmalıdır. Önemli olan husus hırsızlık ihbarı verildiğinde ilgili bölgeye en kısa zamanda ulaşılıp hırsızlık olayına müdahale edilmesidir.

Görsel 5.5'te kablolu hırsız alarm sisteminin 3 katlı bir binaya yerleşimi ve hırsız alarm santrali ile bağlantısı görülmektedir.



Görsel 5.5: 3 katlı bir binada kablolu hırsız alarm sistemi

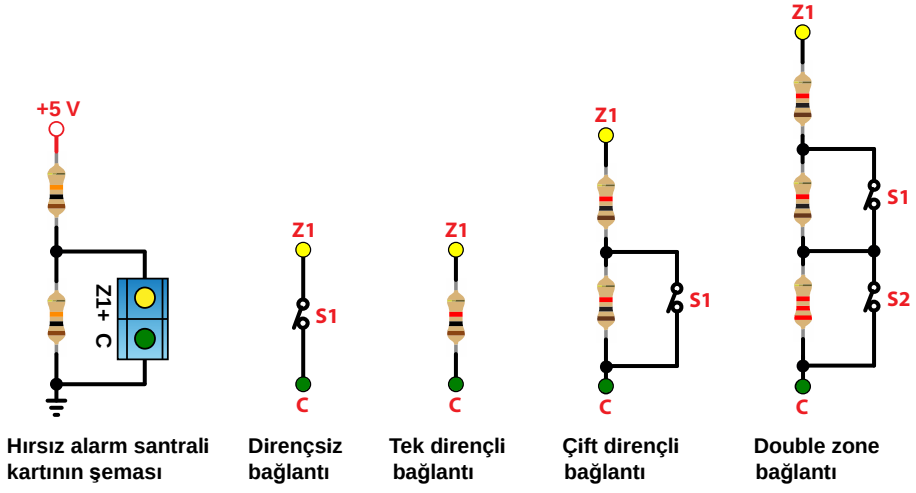
Görsel 5.6'da **4 zone**lu kablolu hırsız alarm sisteminin hırsız alarm santrali ile **zone**lara bağlanacak dedektör ve manyetik kontakların bağlantısı görülmektedir. Aynı zamanda hırsız alarm santraline **keypad** ve siren bağlantıları da yapılmıştır.



Görsel 5.6: 4 **zone**lu kablolu hırsız algılama ve alarm sisteminin bağlantısı

Kablolu hırsız alarm santrallerindeki **zone** yapısı yangın alarm santralindeki **zone** yapısına işlevsellik açısından benzerdir. Yangın alarm santrallerinde **zone**lardaki dedektörlerin değişimleri algılaması akım ölçme prensibine dayanırken kablolu hırsız alarm santrallerinde ise gerilim bölücü

dirençler ile gerilim ölçme mantığına dayanır. Görsel 5.7'de kablolu hırsız alarm santrallerindeki **zone** uçlarının bağlantı mantığının devre şeması görülmektedir.



Görsel 5.7: Kablolu hırsız alarm santrallerindeki zone uçlarının bağlantı mantığının devre şeması

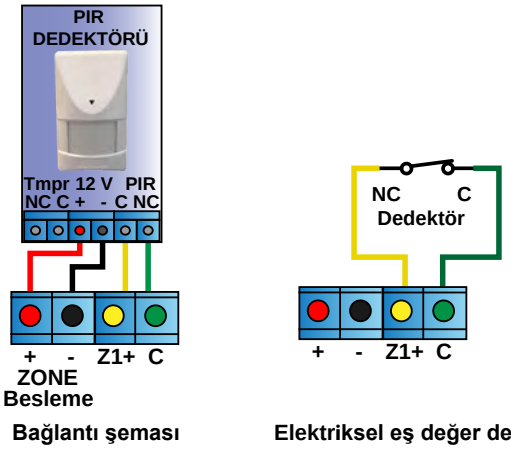
Mikrodenetleyiciler ile programlama konuları işlendikten sonra hırsız alarm santrallerindeki **zone** uçlarının çalışma mantığını anlamak çok basittir. Her bir **zone** ucu, mikrodenetleyicinin analog girişlerine bağlıdır (ADC). Hırsız alarm santrallerindeki ilgili **zone**u aktif etmek demek programcı mantığı ile ilgili analog girişi aktif etmek demektir. İlgili **zone** aktif edildikten sonra analog gerilim ölçümü yapılır. Ölçülen gerilimin değerine göre hırsız alarm santrali denge, hata ve alarm çalışma modlarından birine göre kendini ayarlar.

Zone uçlarına bağlanan dedektör veya manyetik kontakların, aynı röle mantığında olduğu gibi, çıkışlarında NO-C-NC uçları vardır. Bunlar da kuru kontak anahtarlaması prensibine göre çalışır. Genellikle **zone** uçlarına bağlanan dedektörlerin normalde kapalı olan C-NC uçları kullanılır. Bunu sembolize edebilmek için Görsel 5.7'de anahtar kullanılmıştır. Hırsız alarm santrallerinde kullanılan dedektör ve manyetik kontakların genellikle **zone** bağlantı çeşitleri aşağıdaki gibidir:

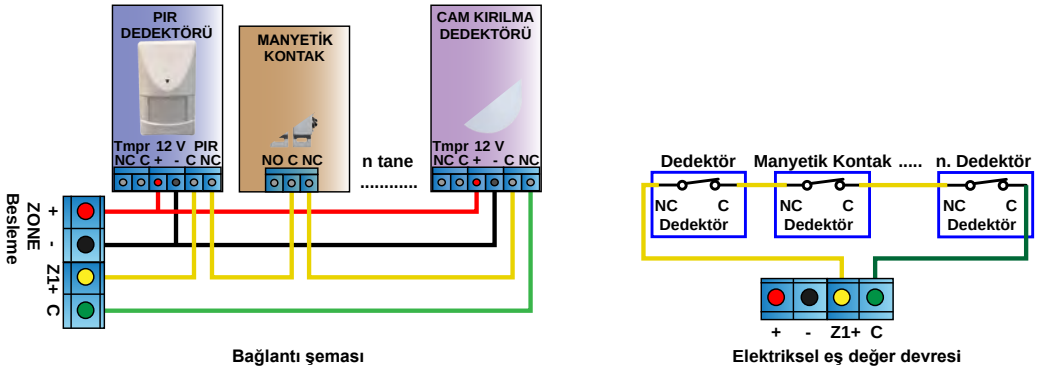
1. Dirençsiz Bağlantı: Hırsız alarm santralindeki **zone** uçlarına dedektör uçlarının direkt bağlanması ile yapılır. Dedektörler genellikle çıkışlarına bağlanan rölelerin kuru kontak anahtarlaması prensibine göre çalışır. Hırsız alarm sistemini bilen biri, hırsız alarm santralinin **zone** ucu ile dedektör bağlantı uçlarının arasına çekilen kablonun üzerinden **zone** uçlarını kısa devre ederek ilgili dedektörü devre dışı bırakılabilir. Bu dezavantajından dolayı çok fazla önerilmemektedir.

a) Bir zonea bir dedektör bağlama: Hırsız alarm santralindeki **zone** uçlarına dedektör uçlarının direkt bağlanması ile yapılır. Görsel 5.8a'da dirençsiz bağlantıda bir **zone**a bir dedektör bağlantısı ve elektriksel eş değer devresi görülmektedir. Elektriksel eş değer devrede; dedektörün bağlantı uçlarının kapalı kontak anahtarının hırsız alarm santrali **zone** uçlarına paralel bağlantısı olarak görülebilir. İlgili dedektör alarm verirse dedektör uçları açık devre olur ve hırsız alarm santralinin **zone** uçları aracılığı ile alarm olarak algılanır.

b) Bir zonea birden fazla dedektör bağlama: Bir **zone**a birden fazla dedektörün çıkışlarının seri bağlanması ile yapılır. Görsel 5.8b'de dirençsiz bağlantıda bir **zone**a birden fazla dedektör ve manyetik kontak bağlantısı ve elektriksel eş değer devresi görülmektedir. Elektriksel eş değer devrede dedektörlerin bağlantı uçlarının kapalı kontak anahtarının kendi aralarında seri bağlantısı, fakat seri bağlı dedektörlerin hepsinin hırsız alarm santrali **zone** uçlarına paralel bağlantısı olarak görülebilir. İlgili dedektör alarm verirse dedektör uçları açık devre olur ve hırsız alarm santralinin **zone** uçları aracılığı ile alarm olarak algılanır.



Görsel 5.8a: Dirençsiz bağlantıda bir **zone**a bir dedektör bağlantısı ve elektriksel eş değer devresi



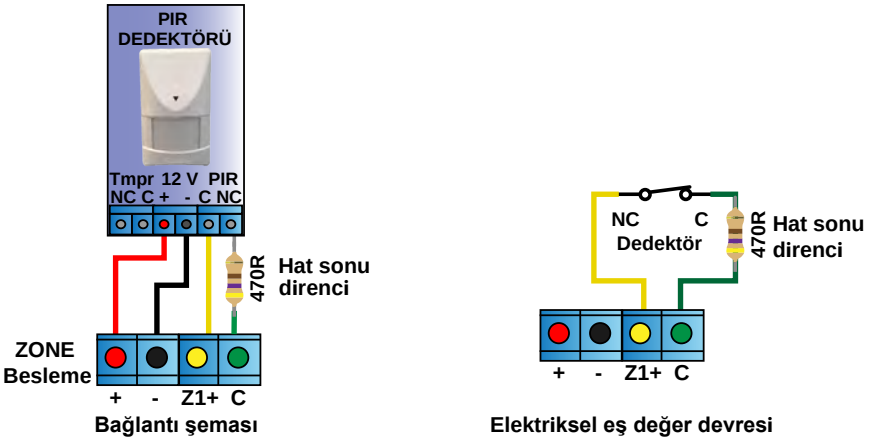
Görsel 5.8b: Dirençsiz bağlantıda bir **zone**a birden fazla dedektör ve manyetik kontak bağlantısı ve elektriksel eş değer devresi

2. Dirençli Bağlantı: Hırsız alarm santralindeki **zone** uçlarına dedektör uçlarındaki hat sonu direncinin bağlanması ile yapılır. Hat sonu direncinin değeri hırsız alarm santralinin markasına ve modeline göre farklılık gösterebilir. En sık kullanılan hat sonu direnç değerlerine 470R, 1K, 2K2, 2K7, 4K7 vb. örnek verilebilir. **Hat sonu direncinin bağlantısı, montaj sırasında mutlaka ilgili dedektörün içine yapılmalıdır. Hırsız alarm santrali içinde zone uçlarına montaj kolaylığı sebebi ile bağlantı yapılmamalıdır.** Tek hat sonu direnci kullanıldığında dedektör kablosundaki bir kopukluk panel tarafından sabotaj olarak algılanabilir. Çift hat sonu direnci kullanıldığında hem dedektör kablosundaki kopukluk hem de kısa devre durumu panel tarafından sabotaj olarak algılanabilir.

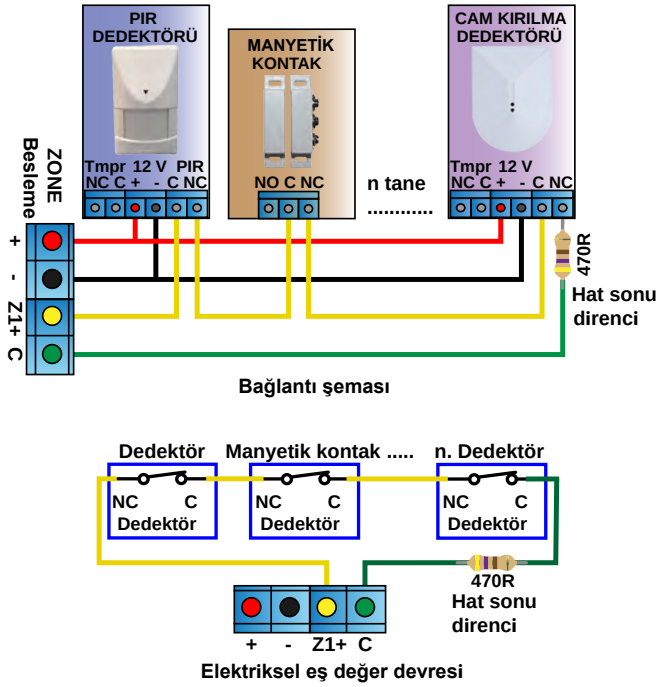
a) Bir zonea bir dedektör bağlama: Hırsız alarm santralindeki **zone** uçlarına dedektör uçlarının hat sonu direncinin seri bağlanması ile yapılır. Görsel 5.9a'da dirençli bağlantıda bir **zone**, bir dedektör bağlantısı ve elektriksel eş değer devresi görülmektedir. Elektriksel eş değer devrede dedektörün bağlantı uçlarının kapalı kontak anahtarının hat sonu direnci ile hırsız alarm santrali **zone** uçlarına paralel bağlantısı olarak görülebilir. İlgili dedektör alarm verirse dedektör uçları açık devre olur ve hırsız alarm santralinin **zone** uçları aracılığı ile alarm olarak algılanır.

b) Bir zonea birden fazla dedektör bağlama: Bir **zone**a birden fazla dedektörün çıkışlarının seri bağlanarak en son dedektöre hat sonu direnci bağlanması ile yapılır. Görsel 5.9b'de dirençli bağlantıda bir **zone**a birden fazla dedektör ve manyetik kontak bağlantısı ile elektriksel eş değer devresi görülmektedir.

Elektriksel eş değer devrede dedektörlerin bağlantı uçlarının kapalı kontak anahtarının kendi aralarında seri bağlantısı, fakat seri bağlı dedektörlerin hepsinin hat sonu direnci ile hırsız alarm santrali **zone** uçlarına paralel bağlantısı olarak görülebilir. İlgili dedektör alarm verirse dedektör uçları açık devre olur ve hırsız alarm santralinin **zone** uçları aracılığı ile alarm olarak algılanır.



Görsel 5.9a: Dirençli bağlantıda bir zone, bir dedektör bağlantısı ve elektriksel eş değer devresi



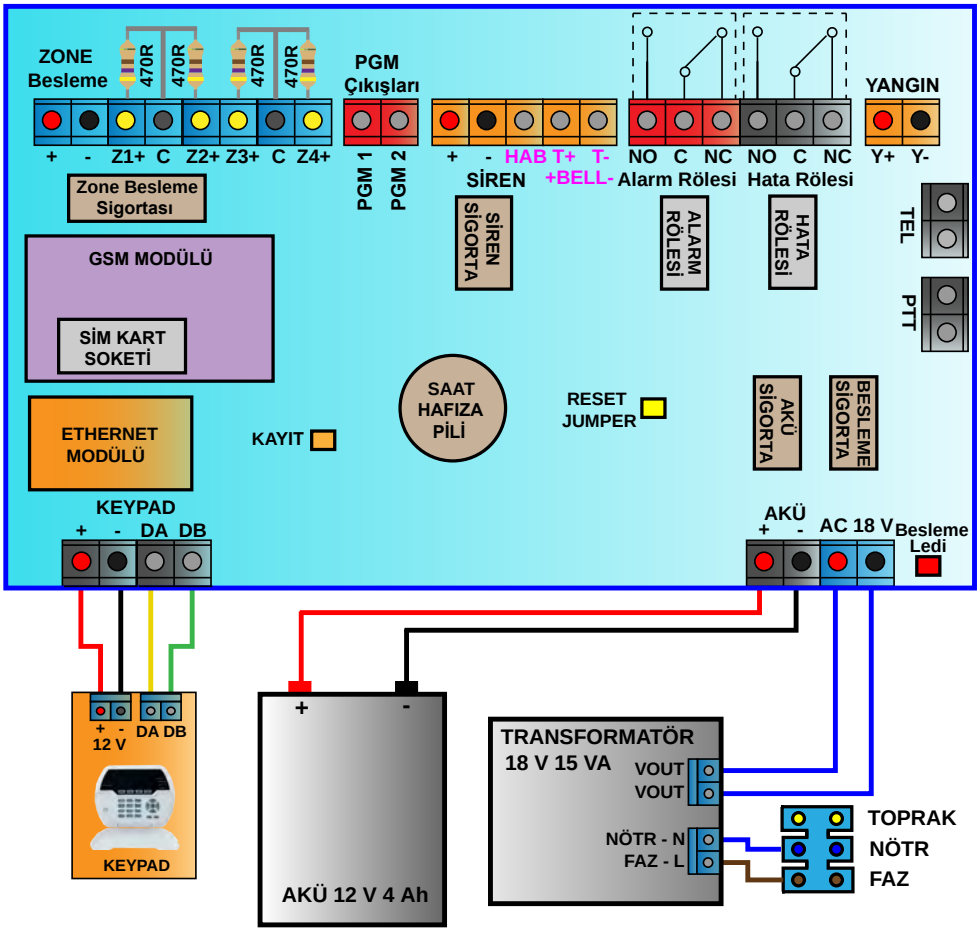
Görsel 5.9b: Dirençli bağlantıda bir **zone**a birden fazla dedektör ve manyetik kontak bağlantısı ve elektriksel eş değer devresi

NOT

Görsellerdeki dirençli bağlantı şemaları farklı marka ve modeldeki hırsız alarm santralleri için farklılık gösterebilir.

Görsel 5.10'da kablolu hırsız alarm santralinin elektronik kartı üzerindeki klemens bağlantıları, güç kaynağı ve akü bağlantıları görülmektedir. Buradaki örnek şemada güç kaynağı olarak transformatör (trafo) kullanılmıştır. Transformatör yerine hazır **switch** mode güç kaynağı kullanılan modelleri de mevcuttur.

Yine Görsel 5.10'da hırsız alarm santral kartının **zone** uçlarının dirençli bağlantısı görülmektedir. Dirençsiz bağlantıda ise **zone** uçlarında direnç yerine **jumper** kullanılabilir ya da boş bırakılabilir. Kullanılan ürünün markası ve modeline göre bağlantılar farklılık gösterebilir. Dirençli ve dirençsiz **zone** bağlantılarının nasıl yapılması gerekiği Görsel 5.8 ve Görsel 5.9'da görülmektedir.



Görsel 5.10: Örnek bir kablolu hırsız alarm santralinin besleme gerilimi ve akü bağlantıları

Kablolu hırsız alarm santrallerini kontrol edebilmek için **keypad** bağlantısının olması gerekir. **Keypad** bağlantısı olmadan hırsız alarm santralinin ilk kurulumunun yapılabilmesi ve çalışması mümkün değildir. **Keypad** bir anlamda hırsız alarm santralinin çalışma durum göstergesi olarak da görev yapar. Yani hırsız alarm santralinin çalışma modlarındaki durumları **keypad** üzerinden takip edilebilir.

Kablolu hırsız alarm santralinin üzerinde genel olarak bulunan çıkış soketleri ve işlevleri aşağıda verilmiştir:

• **Siren bağlantı uçlarına** haricî veya dâhilî siren bağlanabilir. Siren besleme uçları her ihtimale karşılık 0,5 A'lik cam sigorta ile korunmuştur. Fazla akım çekilmesi çıkışa zarar verebilir. Kullanılan marka ve modelin siren klemens uçlarına göre bağlantı yapılır. Kimi sirenler **eksi tetik (T-)** ile tetikleme alırken kimisi de **artı tetik (T+)** ile tetikleme alır. Farklı marka ve modeldeki bütün ürünleri kapsaması için siren klemensleri genel olarak çizilmiştir. Siren bağlantı klemensleri **HAB, T+, T-** olduğu gibi **+BELL-** şeklinde de olabilir. Bu şekilde olan hırsız alarm santrallerinde de tetikleme sinyalleri eksi tetik için **BELL-** bağlantı ucu, artı tetik için **BELL+** bağlantı ucu kullanılır.

• **PGM (Programlanabilir) çıkışları,** herhangi bir seçili olay durumunda aktif hâle gelir. Süreye göre veya seçili olan olayın bitiş durumunda devre dışı olur. Olay durumunda **- tetikleme** verir.

• **Yangın klemensleri,** hırsız alarm sistemlerinde konvansiyonel yangın alarm dedektörlerinin kullanılabilmesi için vardır. Bir anlamda hırsız alarm santrali, yangın bağlantı klemensleri aracılığı ile yangın alarm santralinin kısmi olarak işlevini gerçekleştirir.

• Sistemde hırsız alarm durumu oluştuğunda **hırsız alarm rölesi çıkışı** aktif olmaktadır. Sistemde hata durumu oluştuğunda da **hata rölesi çıkışı** aktif olmaktadır. Alarm veya hata durumu ortadan kalktığında röle eski konumuna dönmektedir. Alarm veya hata rölesi sinyal rölesidir, yüksek akımlı anahtarlama yapılması isteniyorsa arada uygun akımlı röle veya kontaktör kullanılmalıdır.

• **TEL-PTT klemenslerinden** TEL klemensi hırsız alarm santraline telefon bağlamak için kullanılır. PTT klemensi hırsız alarm santrali kullanırken internet de aynı telefon hattından kullanılıyor ise **splitter** aracılığı ile hem telefon hem internet için modemin bağlantısının aynı anda kullanılabilmesini sağlar.

• **GSM modülü,** hırsız alarm santralinin kurulduğu ortamda telefon hattının mümkün olmadığı durumlarda kullanılır. Temel amacı GSM üzerinden hırsız alarm santralinin çalışma durumlarının izlenmesidir. GSM modülünün kullanılabilmesi için SIM kartın yuvasına takılı olması gerektiği unutulmamalıdır.

• **Ethernet modülü,** hırsız alarm santralinin modem üzerinden bağlantısının yapılarak çalışma durumlarının izlenebilmesi için kullanılır.

İlk kurulumda aküler, sisteme şehir şebekesini bağlayıp sistemi çalıştırdıktan sonra bağlanmalıdır. Birincil güç kaynağı bağlanmadan aküler, sistemi çalıştırmayabilir.

5.1.2. Kablosuz Sistem Hırsız Alarm Santrali

Teknolojinin gelişmesi ile birlikte günümüzde kablosuz haberleşme (bağlantı, iletişim) sistemleri de hızlı bir şekilde gelişmektedir. Hırsız alarm sistemlerinde en sık kullanılan kablosuz bağlantı yöntemleri **RF** ve **Wi-Fi**'dir.

RF haberleşme; verilerin kablosuz olarak iletimi için radyo dalgalarının kullanıldığı, bir alıcı ve bir verici ile çok uzak mesafeler arası iletişimin sağlandığı, iletimin çok yüksek hızlara ulaştığı haberleşme yöntemidir. Hırsız alarm sistemlerinde genellikle 433 MHz ve 868 MHz frekansında **RF** iletişimi kullanılmaktadır.

Wi-Fi açılımı [Wireless Fidelity (wayrlîs fıdelıtı)] yani **Kablosuz Bağlantı Alanı** anlamına gelen, kişisel bilgisayar, akıllı telefon, tablet, video oyunu konsolları, dijital ses oynatıcıları ve diğer teknolojik cihazların kablosuz olacak şekilde internete ve birbirlerine bağlanabilmesini sağlayan teknolojidir. **Wi-Fi** bağlantısını destekleyen tüm cihazların, yakınlarında bulunan kablosuz erişim noktaları aracılığıyla yerel alan ağına bağlanabilmelerini sağlar. **Wi-Fi teknolojisi, kablosuz yerel ağ** anlamına gelen WLAN olarak da bilinir. Son zamanlarda IOT teknolojisinin gelişmesi ile birlikte hırsız alarm sistemlerinde yaygın olarak kullanılmaktadır.

Görsel 5.11'de farklı modellerdeki kablosuz hırsız alarm santralleri görülmektedir.



Görsel 5.11: Farklı modellerdeki kablosuz hırsız alarm santralleri

Kablosuz hırsız alarm santralleri, işlevsellik yönünden kablolu hırsız alarm santraline benzer. Kablosuz hırsız alarm santrali de çıkışında kontrol edeceği **zone** sayısına göre isimlendirilir. Aralarındaki en önemli fark ise hırsız alarm santralinin kontrol ettiği ürünlerin besleme gerilimlerinde genellikle pil, bağlantı (iletişim) yönteminde ise kablo yerine **RF, Wi-Fi** gibi kablosuz bağlantı yöntemlerinin kullanılmasıdır.

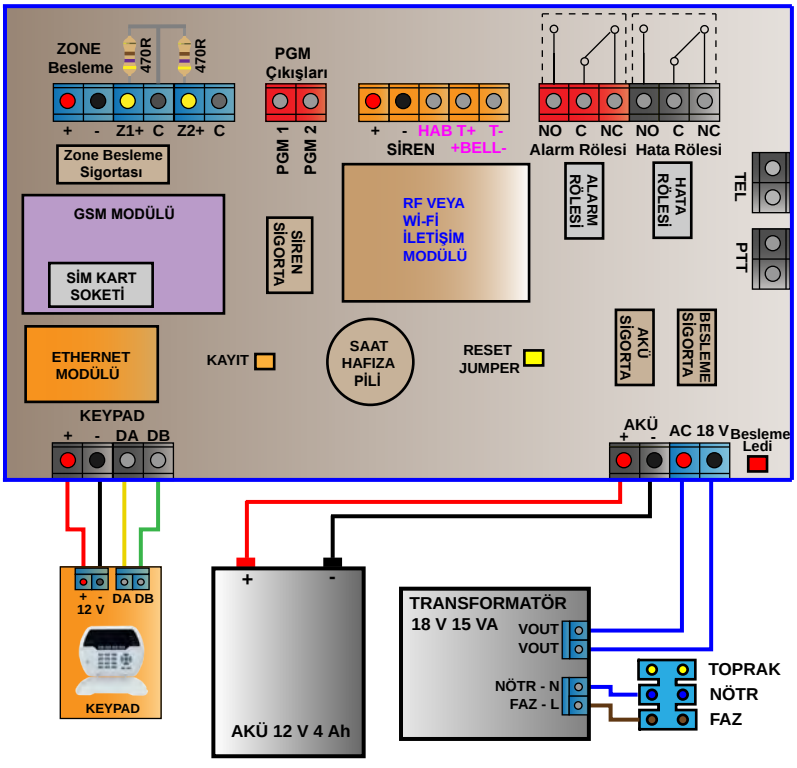
Kablosuz hırsız alarm sistemlerinin montajı kolay olduğu için tercih edilmektedir fakat kullanılan ürünlerin besleme gerilimleri pil ile sağlandığından zaman zaman pillerin değişmesi gerekmektedir. Ancak zaman içinde kullanıcının ürünlerin üzerindeki pili değiştirmeyi unutması bu sistemin olumsuz yönüdür. Başlangıçta montajının kolay olmasının yanında ürünlerin pillerinin sürekli takip gerektirmesi, hırsız alarm sistemlerinin seçiminde göz önüne alınmalıdır.

Kablosuz hırsız alarm sistemlerinde kullanılacak ürünlerin seçiminde, santrale ürünlerin kablosuz tanıtılabilmesi için üzerinde seri numarası (ID No.) olmasına ve bir buton veya **switch**in bulunmasına dikkat edilmelidir.

Kablosuz hırsız alarm sistemlerinde kullanılan bütün ürünlerin santral ile bağlantısı sadece kablosuz (**RF** veya **Wi-Fi** ile) yapılmaz. Her ihtimale karşılık kablolu hırsız alarm santralinde olduğu gibi kablosuz hırsız alarm santralinde de en az 2 adet kablolu **zone** girişi mutlaka bulunur. Bu iki **zone**a önemli yerlere montajı yapılan manyetik kontak veya dedektörler bağlanır.

Görsel 5.12'de kablosuz hırsız alarm santralinin elektronik kartı üzerindeki klemens, güç kaynağı ve akü bağlantıları görülmektedir. Kablolu ve kablosuz hırsız alarm santral kartları arasındaki fark **zone** klemens sayısının azlığı ve yangın klemens bağlantısının olmamasıdır. Sebebi ise kablosuz ürünler kullanılacağı içindir. Geri kalan bağlantı klemensleri ise genel olarak aynıdır. Aynı zamanda kablosuz hırsız alarm santralinin elektronik kartı üzerinde **RF** veya **Wi-Fi** bağlantı modülü de vardır.

Kablosuz hırsız alarm santrallerini kontrol edebilmek için **keypad** bağlantısının olması gerekir. **Keypad** bağlantısı olmadan hırsız alarm santralinin ilk kurulumunun yapılabilmesi ve çalışması mümkün değildir. Burada **keypad** bağlantısı kablolu olarak eklenmiştir. Kablosuz **keypad** de olabilir.



Görsel 5.12: Örnek bir kablosuz hırsız alarm santralinin besleme gerilimi ve akü bağlantıları

5.2. HIRSIZ ALARM SİSTEMİ DONANIMLARI

Hırsız alarm sistemlerinde kullanılan donanımlar, seçilen hırsız alarm santraline göre farklılıklar göstermektedir. Tercih edilen her hırsız alarm santralinde, ister kablolu ister kablosuz sistem olsun, kullanılacak ürünlerin temel çalışma işlevleri aynı olsa da bağlantı ve kurulum farklılıkları vardır. Hırsız alarm sistemlerinde kullanılan ürünlerin, hırsız alarm santraline bağlantısı genellikle 6x0,22 LIY(ST) sheetli kablo ile yapılır. Görsel 5.13'te 6x0,22 LIY(ST) kablonun görseli verilmiştir.



Görsel 5.13: Hırsız alarm sistemlerinde genellikle kullanılan 6x0,22 LIY(ST) kablo

NOT

Hırsız alarm sistemlerinde kullanılan donanımların hırsız alarm santralleri ile aşağıda verilecek bağlantı şemaları marka ve modele göre farklılık gösterebilir.

5.2.1. Keypad

Kablolu veya kablosuz hırsız alarm santrallerini programlamak ve montajı yapılan hırsız alarm sisteminin arıza ve alarm durumlarını görebilmek için kullanılır. **Keypad** ile hırsız alarm santraline şifre girme, hırsız alarm sistemini kurma ve kurulan sistemi kapatma işlemleri yapılabilir.

Görsel 5.14'te kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki **keypad**lerin görselleri görülmektedir.



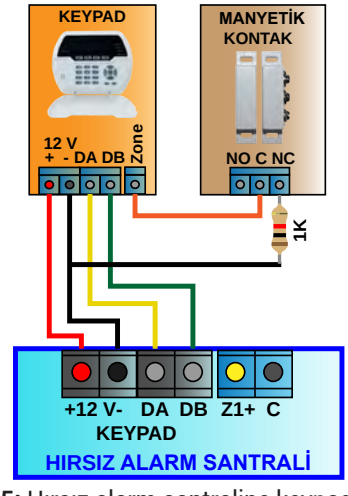
Görsel 5.14: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz keypadler

Keypad, hırsız alarm santralinin programlanması ve yönetilmesini sağlar. **Keypad** montajı yapılırken **keypad**in yerden yüksekliği kullanacak olan kişilerin boy ortalamasına göre değişim gösterir. Yerden yüksekliği ortalama 150 cm gibi düşünülebilir.

Kablolu **keypad**in hırsız alarm santraline bağlantı şeması görsel 5.15'te görülmektedir. Görsel 5.10 ve Görsel 5.12'den farkı **keypad**in üzerinde **zone** ucunun olmasıdır. Bu bağlantı ucu, genellikle kapı girişlerine konan **keypad**in aynı kapıya ekstra manyetik kontak bağlantısı için kullanılır. Bu **zone** genellikle giriş gecikmeli olarak programlanır. Hırsız alarm sisteminin monte edildiği yerde birden fazla kapı var ise birden fazla **keypad** bağlantısı (genellikle 4-15 adet) hırsız alarm santraline yapılabilir.

Görsel 5.15'te **keypad**in besleme (12 V, GND) uçlarının ve data (DA, DB) uçlarının hırsız alarm santralindeki ilgili uçlara bağlantısı görülmektedir. Data uçlarının DA ve DB olması **keypad** ile hırsız alarm santrali arasında kullanılan iletişim protokolünün RS 485 olduğunun göstergesidir. RS 485 kullanılmasının sebebi ise iki ürün arasında bağlantı için kullanılan kablonun boyunun uzun olmasıdır. Maksimum 76 m (UL standardına göre) tavsiye edilir.

Hırsız alarm santrali üzerindeki ve **keypad**deki data uçlarının isimleri kullanılan modele göre farklılık gösterebilir.



Görsel 5.15: Hırsız alarm santraline keypad bağlantısı

5.2.2. PIR (Pasif İnfrared) Dedektörü

PIR sensörleri, ortamdaki sıcaklık ve kızılötesi dalga değişimlerine göre hareket algılayan sensörlerdir. Çalışma prensipleri ise şu şekildedir: Tüm nesneler bulundukları ortama sıcaklık ve kızılötesi dalga yayar. PIR sensörlerin yapısında bir fresnel lens vardır. Fresnel lensler, ışık toplamak ve gelen ışığı kesintili oluşmasını sağlamak için kullanılan taşınabilir ve hafif tabakalardır. Bu mercek sayesinde ortamdaki ışınlar sensörün tam üzerine düşüp odaklanmayı sağlar. Sensör, sabit olan sıcaklık ve kızılötesi dalgalar değiştiğinde bunu algılayarak sisteme bildirir. Örneğin bir odaya monte edilen PIR sensörü, odadaki duvarın ve cisimlerin yaydığı sabit dalga değerlerini kontrol eder. Odaya bir insan girdiğinde bu değerler değişime uğrayacağından sensör algılar ve bağlı olduğu sisteme bilgi verir. Görsel 5.16'da analog ve dijital PIR sensörlerinin görselleri görülmektedir.



Görsel 5.16: Analog ve dijital PIR sensörleri

PIR sensörleri genel olarak hareket algılama özelliği açısından analog ve dijital PIR sensörleri olarak gruplandırılabilir. Analog PIR sensörleri hareketi algılar ve çıkışında analog gerilim değeri üretir. Dijital PIR sensörleri ise hareketi algılar ve çıkışında dijital sinyal üretir.

PIR sensörleri, ürettikleri sinyalleri kontrol edecek elektronik devre ile kullanıldığında **PIR dedektörü** olarak isimlendirilir. PIR dedektörü hareket dedektörü olarak da bilinmektedir. Genellikle hırsız alarm sistemlerinde kullanılan PIR dedektörlerinde ister analog ister dijital PIR sensörü kullanılmış olsun her ikisinde de hareket algılama sistemi ve kontrol edecek elektronik devre yapısı farklı olurken çıkış bağlantı klemensleri aynı yapıdadır. Yani kullanılan sensörün özelliğine göre PIR dedektörleri de analog ve dijital PIR dedektörleri olarak gruplandırılır.

PIR dedektörleri genel olarak canlı varlıkların hareket hâlinde iken yaydığı kızılötesi enerjiyi algılama prensibine göre çalışır ve algılama sırasında hiçbir ışın yaymaz. İsteğe bağlı olarak hayvan algılamayan modelleri mevcuttur.

PIR dedektörlerlerinin montajını yaparken aşağıdaki maddelere dikkat edilmelidir:

• PIR dedektörler 2,1 m (7 ft) yükseklikte optimal performans gösterecek şekilde tasarlanmıştır fakat daha yükseğe veya aşağıya monte edilebilir. İstenen yükseklik arka plastik taraftaki işaretlere göre ayarlanabilir.

• PIR dedektörü; yansıma yüzeylerinin yakınına, direkt hava gelen vantilatör, fan ve pencereler, yağ ve buhar vakumları, kızılötesi ışık kaynakları, ısıtıcı gibi hava değişikliklerine neden olacak cihazların yanına, buzdolapları ve fırınların yakınına monte edilmemelidir.

• Evcil hayvanın üstüne tırmanabileceği 0,9 m (3 ft) üzeri nesne veya mobilya (örneğin kedi koltuk üzerinde) dedektörün 2,1 m (7 ft) alanı içine konulmamalıdır. Ayrıca dedektör, evcil hayvanın geçiş yapabileceği merdivene monte edilmemelidir.

• PIR dedektörünün içindeki sensörün yüzeyine dokunulmamalıdır. Dedektörün kötü çalışmasına sebep olabilir. PIR sensörü, saf alkollü yumuşak bir bezle temizlenebilir.

• Dedektörün bir hayvanın bağışıklığına göre yanlış alarm vermesi hayvanın büyüklüğüne, ısısına ve kürk kalınlığına göre değişebilir.

• PIR dedektörün montajı yapılırken mutlaka köşe noktalara montajlanmasına özen gösterilmelidir. Kapının ve pencerenin karşısına gelecek şekilde montaj yapmak hatalı olacaktır.

• PIR dedektörünü test etmek için dedektörce korunan alan boydan boya yürünür. Eğer yeşil LED 3 saniye boyunca yanıyorsa dedektör çalışıyor demektir. Hayvan bağışıklığı testi için ise evcil hayvan algılama bölgesinde yürütülür. Eğer ışık yanmıyorsa hayvan algılamaya karşı bağışıktır. Yürüme testi sırasında koruma alanı daima boydan boya geçilmelidir. Dedektöre doğru yürünmemelidir.

PIR dedektörleri aşağıdaki gibi sınıflandırılır:

a) PIR Dedektörü: Normal analog PIR hareket dedektörleridir. Her türlü hareketin algılanmasını sağlar. Görüş açısı 110 derecedir. İç ve dış ortamlarda kullanılabilecek modelleri vardır. Görsel 5.17'de kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki PIR dedektörlerinin görselleri görülmektedir.



Görsel 5.17: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz PIR dedektörleri

b) PET Dedektörü: Hayvan algılamayan PIR dedektörlerine **PET dedektörü** denir. PET dedektörü, genel olarak ağırlığı 15-40 kg kadar olan nesnelerin hareketini görmezden gelir. Görüş açısı 90 derecedir. İç ve dış ortamlarda kullanılan modelleri vardır. Görsel 5.18'de kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki PET dedektörlerinin görselleri görülmektedir.



Görsel 5.18: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz PET dedektörleri

c) Perde Tipi PIR Dedektörü: Genellikle pencere ve sürgülü kapılarda kullanılır. Görüş açısı 0 ile 10 derece arasında ayarlanabilir. Görsel 5.19'da kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki perde tipi PIR dedektörlerinin görselleri görülmektedir.



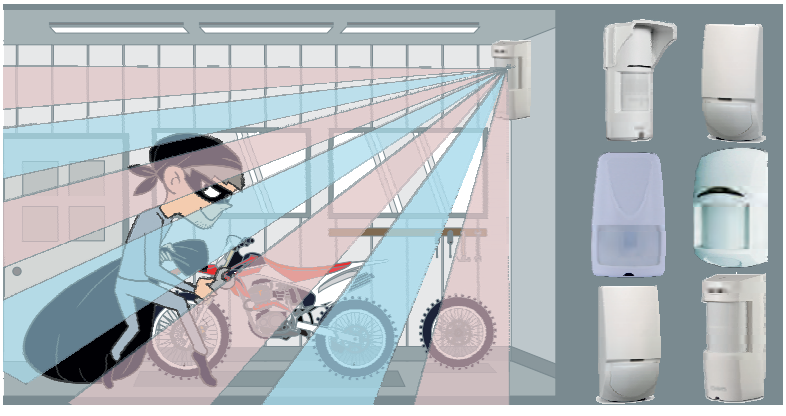
Görsel 5.19: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz perde tipi PIR dedektörleri

ç) Tavan Tipi PIR Dedektörü: Genellikle tavana monte edilir. Görüş açısı 360 derecedir. Görsel 5.20'de kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki tavan tipi PIR dedektörlerinin görselleri görülmektedir.



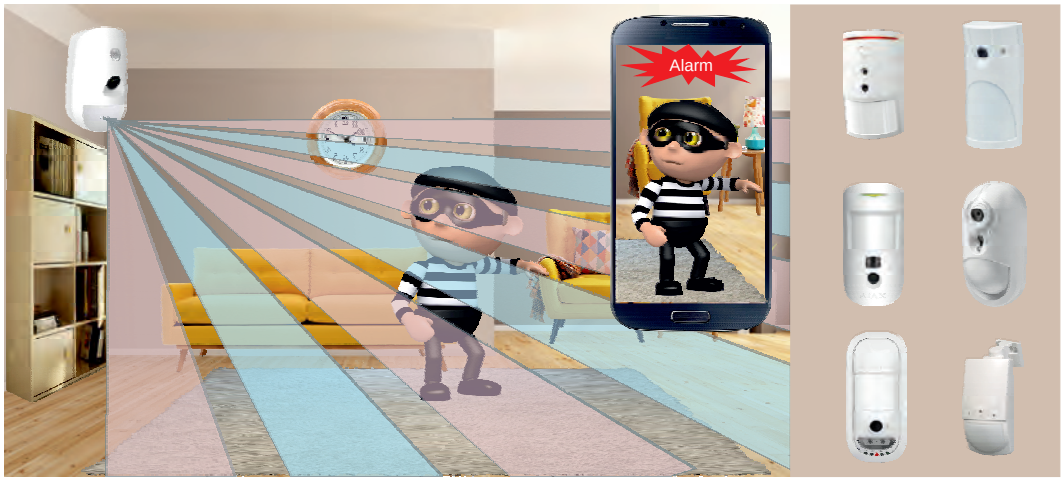
Görsel 5.20: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz tavan tipi PIR dedektörleri

d) Mikrodalga PIR Dedektörü: Hassas algılama istenen yerlerde kullanılır. Hem mikrodalga sinyal ile hem de kızılötesi sinyaller ile algılama yapabilir. Dedektörün algılama alanının önüne objeler koyulması, lens kısmının boyanması gibi etkenler mikrodalga PIR dedektörünün çalışmasını engellemez. Görüş açısı 90 derecedir ve mesafe ayarı yapılabilir. İç ve dış ortamlarda kullanılan modelleri vardır. Görsel 5.21'de kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki mikrodalga PIR dedektörlerinin görselleri görülmektedir.



Görsel 5.21: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz mikrodalga PIR dedektörleri

e) Kameralı PIR Dedektörü: Uzaktan hırsız alarm sistemini görsel olarak da izleme imkânı sağlar. Kameralı PIR dedektörleri, hassas algılama istenen yerlerde kullanılır. Kameralı PIR dedektörleriyle 7/24 takılı olduğu ortam sesli ve HD görüntülü olarak izlenebilir, alarm durumunda video doğrulaması ile alarmın neden çaldığı görülebilir. Alarm öncesi 3 sn.den başlayıp alarm süresince oluşan video görüntüsünü uygulama üzerinden kullanıcılara gönderir. Ortam izlemesi yapılırken anlık olarak uygulama üzerinden kayıt yapma özelliğine sahiptir. İç ve dış ortamlarda çalışabilen modelleri mevcuttur. Görsel 5.22'de kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki kameralı PIR dedektörlerinin görselleri görülmektedir.



Görsel 5.22: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz kameralı PIR dedektörleri

5.2.3. Cam Kırılma Dedektörü

Camın kırılırken çıkardığı sese duyarlı elektronik dedektördür. Genellikle bir oda içinde cama yakın bir noktaya monte edilir. Bir adet cam kırılma dedektörü bütün odadaki camları korumaya yetebilir. Cam kırılma dedektörü; ses, darbe ve şok dalgası analizi yapabilir. Görsel 5.23'te kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki cam kırılma dedektörlerinin görselleri görülmektedir.



Görsel 5.23: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz cam kırılma dedektörleri

Cam kırılma dedektörünün montajı yapılırken montaj yeri tavan veya duvarda korunacak cama bitişik veya karşılıklı olarak seçilmelidir. Montaj, algılama açısına uyacak şekilde ayarlanmalıdır. Dedektörün mikrofon kısmının direkt cam tarafına bakması gereklidir. Korunacak camla arasında herhangi bir engel olmamalıdır. Dedektörün pozisyonu korunacak cama optimal açıda

olmalıdır. Zil, çan, fan, kompresör ve gürültülü makinelerin yakınına dedektörü monte etmekten sakınılmalıdır.

5.2.4. Darbe Dedektörleri

Darbe dedektörleri yüksek hassasiyetli piezoelektrik sensörler kullanılarak imal edilir. Genellikle yüksek güvenlik gerektiren yerlerin duvarlarında ve kasaların yüzeyinin korunmasında kullanılır. Görsel 5.24'te kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki darbe dedektörlerinin görselleri görülmektedir.



Görsel 5.24: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz darbe dedektörleri

Darbe dedektörlerinin montajı yatay veya dikey olarak yapılır. Kasa koruyucunun yakınında elektrik motorları, trafolar, havalandırmalar ve klimalar olmamalıdır. Montaj yapılan yüzeyin üzerinde 2,5 m çapında koruma alanı oluşturur. Montaj yapılan duvarın yüzeyine bağlı olarak birden fazla darbe dedektörü kullanılması gerekebilir. Sismik sarsıntı, vibrasyonlu ortam gibi yoğunluk olan yerlerde (tramvay hattı, yüksek sesli eğlence mekânlarının yanı) montaj yapılır ise hassasiyet değerlerinin hatalı alarm vermemesi için mutlaka düşürülmesi gerekir. Montaj yapılan yerin hassasiyet durumuna göre 24 saat **zone** tipi veya ani **zone** tipi olarak programlanmalıdır.

5.2.5. Su Basma Dedektörleri

Su basma dedektörleri, kaçak su sızıntılarını veya su baskınlarını algılamak için kullanılır. Su sızıntılarının olduğu yerlerin (boru tesisatı, buzdolapları, çamaşır makineleri, su ısıtıcıları, tuvaletler, merdivenler, garajlar vb.) yanına monte edilerek muhtemel olarak meydana gelebilecek su baskınlarına karşı koruma sağlar. Görsel 5.25'te kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki su basma dedektörlerinin görselleri görülmektedir.



Görsel 5.25: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz su basma dedektörleri

Su basma dedektörlerinin algılama uçları sensör üzerindeki plastik kızak altında bulunmaktadır. Plastik kızak 9 cm kaydırılıp uzatma kablosu ile beraber 45 cm uzatma yapılarak (seçenekli) montaj yapılabilir. Su baskını dedektörünün sensörü yer yüzeyinden 1-2 cm yukarıya, su baskını dedektörünün ise yerden 45 cm yukarıya montajı yapılmalıdır. Alarm sistemi kurulu olsun veya olmasın devrede olmalıdır. 24 saat **zone** tipi olarak programlanmalıdır.

5.2.6. Dedektör Bağlantıları

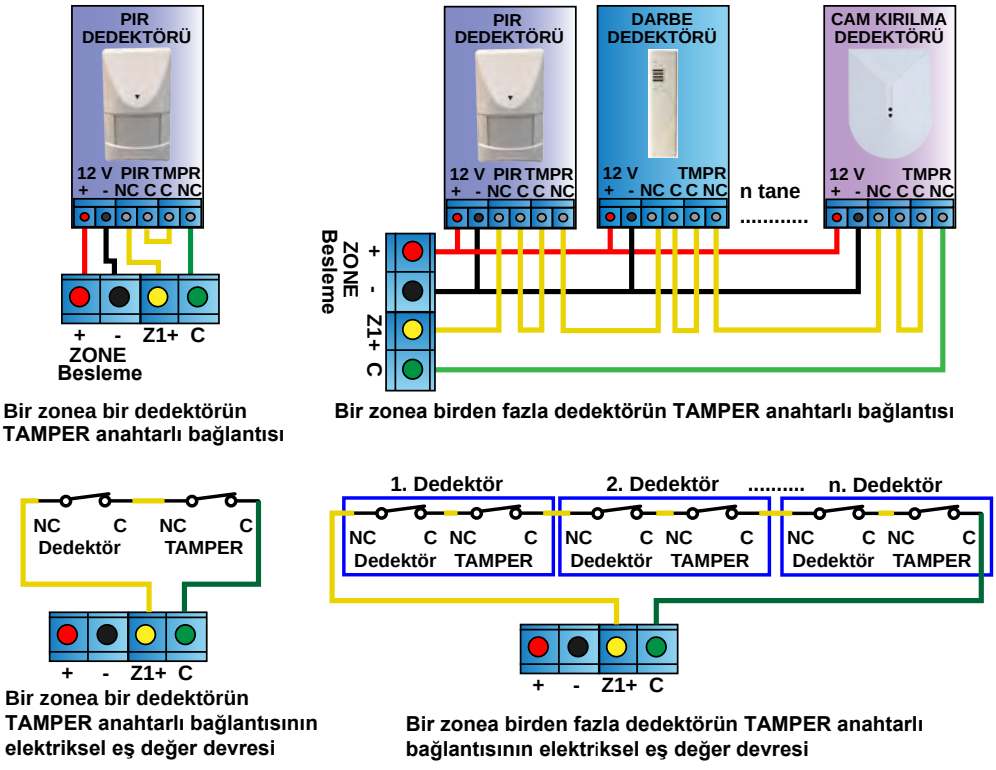
Hırsız alarm sistemlerinde kullanılan dedektörlerin bağlantıları genellikle besleme gerilimleri ve kapalı kontaklar üzerinden yapılır. Dedektör bağlantıları yapılırken **TAMPER** isminde ekstra iki bağlantı ucu bulunur. **TAMPER** bağlantı uçları, hırsız alarm sistemlerindeki bütün ürünlerin kapaklarının altındaki **switch**i ifade eder. Herhangi bir şekilde dedektörün kapağı açılır ise sabotaj alarmı vermek için kullanılır. Aynı dedektör çıkışında bulunan kapalı kontak (C-NC) gibi çalışır. Dedektörün kapağı kapalı iken **TAMPER** kapalı devre, açıldığında ise açık devre gösterir. Bu özelliğinden dolayı kapak veya sabotaj anahtarı olarak bilinir.

Hırsız alarm sistemlerinde kullanılan bütün ürünlerin temel bağlantı mantığı aynıdır. Seçilen ürünün besleme gerilimi var ise bağlantısı yapılır. Kontak uçlarından kapalı kontak uçları (NC-C) kullanılır. Sabotaj anahtarı bağlantısı yapılacak ise **TAMPER** uçları kullanılır. Dikkat edilirse seçilen ürünün hırsız alarm santraline bağlantısı yapılırken en çok 6 adet kabloya ihtiyaç duyulur. Bunun için genellikle 6x0.22 data kablosu kullanılır.

Hırsız alarm sistemlerinde kullanılan dedektörlerin hırsız alarm santraline bağlantısı hat sonu direnç (end off line) değeri hariç çok benzerdir. Tercih edilen ürünün katalog bilgilerine göre burada verilen bağlantı şemaları mutlaka kontrol edilmelidir.

Hırsız alarm sistemlerinde dedektörlerin **zone** uçlarına bağlantısı aşağıdaki yöntemler ile yapılır. Dikkat edilmesi gereken ise **TAMPER** anahtarının kapalı kontaklarının (NC-C) hatta seri bağlanmasıdır. Bunun için genellikle hırsız alarm sistemi üreticileri, dedektör üzerindeki kontak uçları ile **TAMPER** anahtarının uçlarının bir tanesinin elektronik kart üzerinde bağlantısı yapılmış olarak imal ederler. Görsel 5.26, Görsel 5.27, Görsel 5.28 ve Görsel 5.29'da bu bağlantının yapılmadığı düşünülerek bağlantı şemaları oluşturulmuştur.

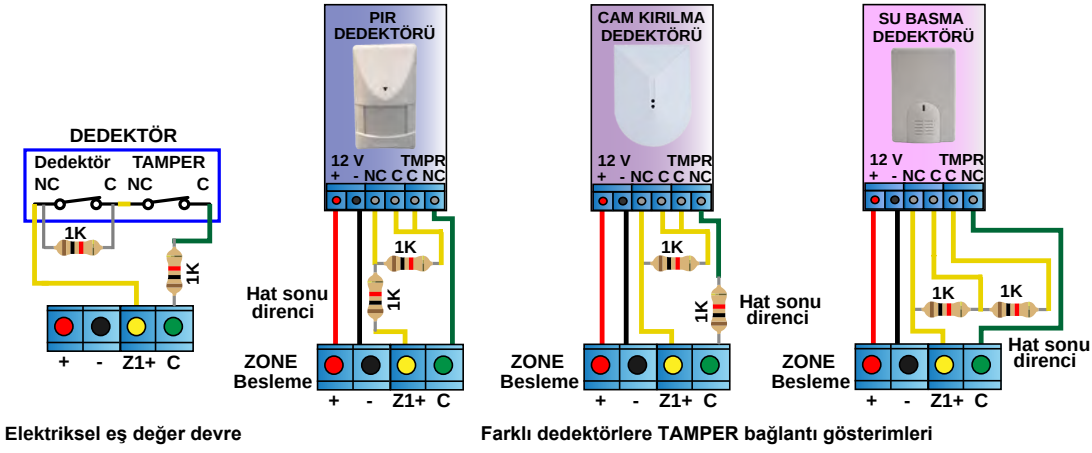
1. Dirençsiz Bağlantı: Görsel 5.26'da hırsız alarm santralinin **zone** uçlarına dedektörlerin **TAMPER** anahtarlı dirençsiz bağlantısı ve elektriksel eş değerleri görülmektedir. Burada hırsız alarm santralindeki besleme gerilimi bağlantı klemensleri, kullanılan dedektörlerin besleme gerilimi bağlantı klemenslerine kırmızı-siyah kablo kullanılarak yapılır. **Zone** uçlarının bağlantısı dedektör kontak uçlarına **TAMPER** anahtar uçları seri bağlanarak sarı ve yeşil kablolarla yapılır. Aynı zamanda dedektörlerin çalışma mantığının daha rahat algılanabilmesi için elektriksel eş değer devreleri de verilmiştir. Bu elektriksel eş değer devre mantığı bütün hırsız alarm sistemlerindeki ürünler için düşünülebilir. Dikkat edilecek olursa dedektörlerin bağlantı uçlarının kapalı kontak anahtarların seri bağlantısı olduğu görülür. İlgili dedektör alarm verirse dedektör uçları açık devre, ilgili dedektörün kapağı açılır ise **TAMPER** (sabotaj) anahtarı açık devre olur. Her iki durumda da hırsız alarm santralinin **zone** uçları aracılığı ile alarm olarak algılanır.



Görsel 5.26: Hırsız alarm santralinin zone uçlarına dedektörlerin TAMPER anahtarlı dirençsiz bağlantısı

2. Dirençli Bağlantı: En çok tercih edilen bağlantı türüdür. Kullanılan ürünün markasına ve modeline göre bağlantı şeması farklılık gösterebilir. Görsel 5.27, 5.28, 5.29a ve 5.29b'deki hat sonu dirençlerinin değerlerinin ve direnç bağlantılarının Görsel 5.9'dakinden farklı olduğu görülmektedir. **Hat sonu direncinin montaj sırasında mutlaka ilgili dedektörün içine bağlantısı yapılmalıdır. Hırsız alarm santrali içinde zone uçlarına montaj kolaylığı sebebi ile bağlantısı yapılmamalıdır.** Tek hat sonu direnci kullanıldığında dedektör kablosundaki bir kopukluk panel tarafından sabotaj olarak algılanabilir. Çift hat sonu direnci kullanıldığında hem dedektör kablosundaki kopukluk hem de kısa devre durumu panel tarafından sabotaj olarak algılanabilir.

a) Bir zone, bir dedektör bağlantısı: Görsel 5.27'de bir **zone**a bir dedektör bağlantısının elektriksel eş değer devresi ve aynı bağlantı şemasının farklı şekilde çizimleri görülmektedir. **Zone** uçlarına bağlanan hat sonu direnç değerine göre hırsız alarm santrali çalışma modunu belirler. **Zone** uçlarındaki direnç değeri **1K** ise hırsız alarm santrali denge modunda çalışır. Denge modunda dedektörün NC-C kontakları kısa devredir. NC-C kontaklarına paralel bağlanan **1K** direncin değeri önemsizdir. **Zone** uçlarında **1K** direnç değeri görünür. **Zone** uçlarındaki direnç değeri **2K** ise hırsız alarm santrali alarm modunda çalışır. Alarm modunda dedektörün NC-C arası açık devre olur. NC-C uçlarına paralel olan **1K** direnç devreye seri bağlanmış olur. **Zone** uçlarında ise **1K+1 K=2K** direnç değeri görünür. **Zone** uçlarındaki direnç değeri kısa devre (0R) ise hırsız alarm santrali hata modunda çalışır. **Zone** uçlarındaki direnç değeri açık devre (sonsuz direnç) ise hırsız alarm santrali yine alarm modunda çalışır. Bu durumda **TAMPER** (sabotaj) anahtarının açık olduğu veya bağlantı hatası olduğu anlaşılır.



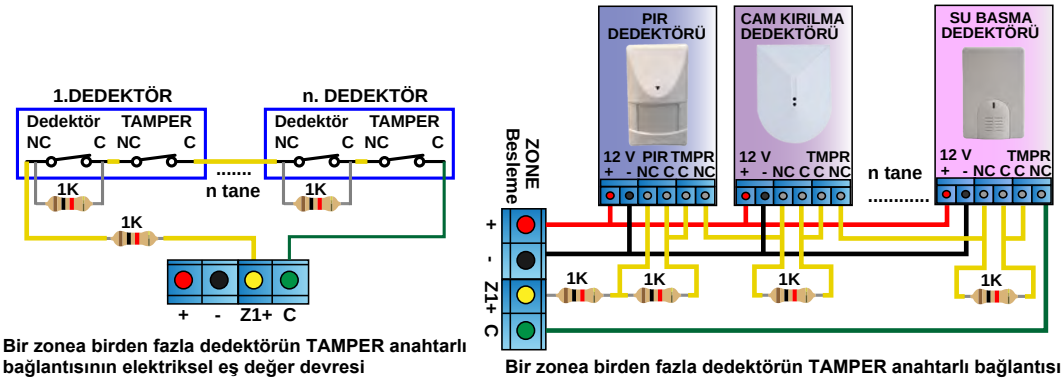
Görsel 5.27: Bir **zone**a bir dedektör bağlantısının **TAMPER** anahtarlı ve dirençli bağlantısı

b) Bir zonea birden fazla dedektör bağlantısı: Görsel 5.28'de bir **zone**a birden fazla dedektör bağlantısının elektriksel eş değer devresi ve bağlantı şemasının çizimleri görülmektedir. Bir **zone**a bir dedektör bağlamak ile birden fazla dedektör bağlamak arasında sistemin çalışma mantığı açısından fark yoktur çünkü kapalı durumdaki anahtarın seri bağlantı yöntemi ile eklenmesi ile seri devreye bağlantısı yapılır. Bu bağlantı yapısında hırsız alarm santralinin çalışma modunu belirleyecek olan hat sonu direnç değeridir. Bu bağlantının avantajları ve dezavantajları aşağıdaki gibidir:

• Hırsız alarm sisteminin montajında hırsız alarm santralinin **zone** sayısı ile kullanılacak dedektör sayısının yetmediği durumlarda kullanılır. Az **zone** sayısına sahip hırsız alarm santrali ile fazla dedektör kontrolü sağlanabilir.

• Hırsız alarm sisteminin montajında hırsız alarm santrali ile dedektörler arasındaki bağlantının kablolaması sırasında fazla kablo çekmemek için kullanılabilir.

• Bir **zone**a birden fazla dedektör bağlandığı için hırsız alarm santralinin alarm durumunda hangi dedektörden alarm geldiğinin tespitinde sıkıntı yaşanabilir. En önemli dezavantajıdır.



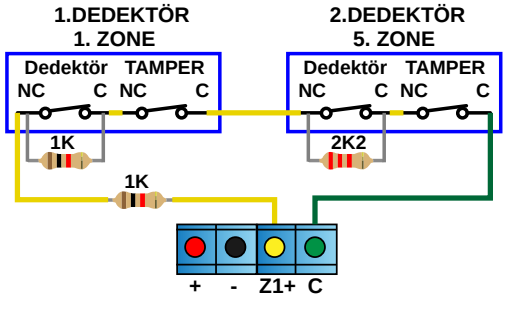
Görsel 5.28: Bir **zone**a birden fazla dedektör bağlantısının elektriksel eş değer devresi ve **TAMPER** anahtarlı ve dirençli bağlantı şeması

c) Double zone (zone çiftleme) bağlantısı: Double zone özelliği ile bir **zone** iki ayrı **zone** olarak kullanılabilir. Bu bağlantı çeşidini her hırsız alarm santrali desteklemez. **Zone** uçlarına bağlanan hat sonu direnç değerine göre hırsız alarm santrali çalışma modunu belirler.

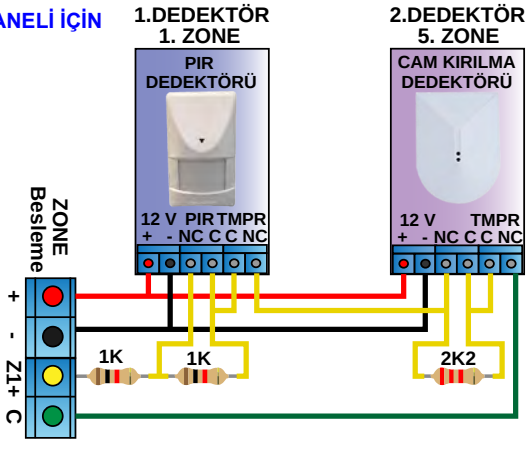
Görsel 5.29a'da **double zone** bağlantısının **TAMPER** anahtarlı bağlantısının elektriksel eş değer

devresi görülmektedir. **Zone** uçlarındaki direnç değeri **1K** ise hırsız alarm santrali denge modunda çalışır. Denge modunda dedektörün NC-C kontakları kısa devredir. NC-C kontaklarına paralel bağlanan **1K** ve **2K2** dirençlerinin değeri önemsizdir. **Zone** uçlarında **1K** direnç değeri gözükür. **Zone** uçlarındaki direnç değeri **2K** ise hırsız alarm santrali normal **zone**dan aldığı sinyal ile alarm modunda çalışır. Alarm modunda dedektörün NC-C arası açık devre olur. NC-C uçlarına paralel olan **1K** direnç devreye seri bağlanmış olur. **Zone** uçlarında ise **1K+1K=2K** direnç değeri görünür. **Zone** uçlarındaki direnç değeri **3K2** ise hırsız alarm santrali **double zone**dan aldığı sinyal ile alarm modunda çalışır. Alarm modunda double bağlanan dedektörün NC-C arası açık devre olur. NC-C uçlarına paralel olan **2K2** direnç devreye seri bağlanmış olur. **Zone** uçlarında ise **1K+2K2=3K2** direnç değeri görünür. **Zone** uçlarındaki direnç değeri kısa devre (0R) ise hırsız alarm santrali hata modunda çalışır. **Zone** uçlarındaki direnç değeri açık devre (sonsuz direnç) ise hırsız alarm santrali yine alarm modunda çalışır. Bu durumda **TAMPER** anahtarının açık olduğu veya bağlantı hatası olduğu anlaşılır.

Görsel 5.29b'de 4 **zone**lu hırsız alarm santralinin 1. **zone**nunu hem normal 1. **zone** olarak kullanımının hem de 5. **zone** olarak kullanımının bağlantı şeması görülmektedir. **Double zone** demek bir **zone**a birden fazla **zone** numarası vermek demektir. Örneğin 4 **zone**a sahip bir hırsız alarm santralinde **double zone** desteği varsa bir **zone**a iki dedektör paralel bağlanır. Bu dedektörlerden birincisine normal hat sonu direnç değeri bağlanırken ikinci dedektöre ise **double zone** hat sonu direnç değerinde direnç bağlanır. Böylece bir **zone** bir anlamda iki **zone** olarak görev yapar. Fakat **keypad**deki gösterge ışığında 1. **zone** 1'de yanarken **double zone**nun ışığı 5'te yanar. Yani en sonuncu **zone**dan bir sonraki **zone** numarasını alır. Böylelikle **double zone** özelliği ile 4 **zone**lu bir hırsız alarm santrali 8 zone olarak kullanılabilir. Aynı zamanda **double zone** özelliği hırsız alarm santralinde **zone** genişleme kartının yaptığı görevi de yapar.



Görsel 5.29a: Bir zonenun double zone bağlantısının elektriksel eş değeri



Görsel 5.29b: Bir zonenun double zone bağlantısının şeması

5.2.7. Beam (Işın) Dedektörleri

Beam dedektörleri alıcı ve verici olmak üzere iki üründen oluşur. Alıcı ve verici birbirini görecek şekilde karşılıklı yerleştirilir. Araya bir nesne girmesiyle verici ünitenin yaydığı ışık alıcı üniteye gidemez. Bu sırada alıcı ünitenin çıkışındaki röle konum değiştirir. Hırsız alarm santrali tarafından alarm bilgisi olarak algılanır. Dâhilî (iç mekân) ve haricî (dış mekân) olarak kullanılan beam dedektörleri vardır.

Beam dedektörleri genellikle pencere pervaz içlerine, giriş kat pencerelerinin dış kısmına, binaların dış çevrelerine, tesislerin sınır duvarlarına, tel ve çitlerin üzerlerine yerleştirilerek haricî güvenliği sağlar.

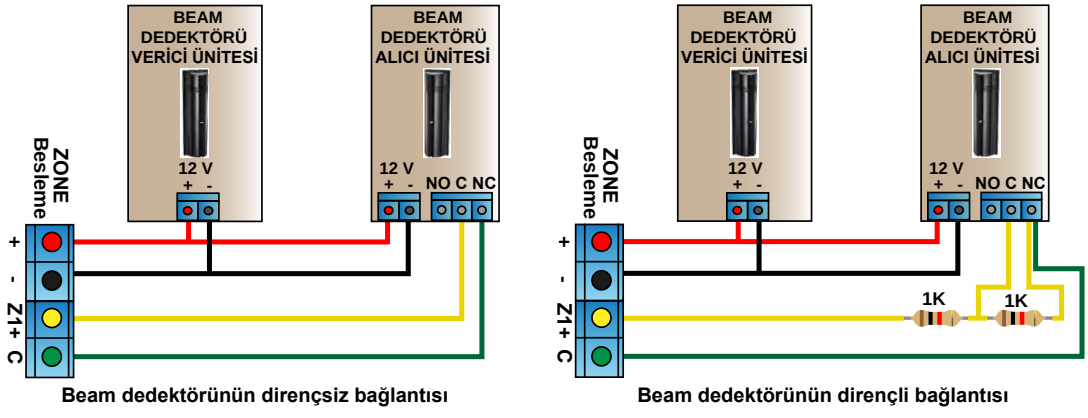
Beam dedektörlerinin montajı yapılırken yönünün güneş ışığı, otomobil farı vb. ışık kaynaklarına direkt gelebilecek şekilde ayarlanmaması gerekir. Optik eksende uzun süre kalabilecek bir ışık, bozulmaya neden olmaz ama cihazın ömrünü etkiler. Cihaz, kirli suya veya deniz serpintilerine maruz kalabilecek yerlere ve sabit olmayan (sallanan) yüzeylere monte edilmemelidir. Yerden minimum 40 cm yüksekliğe montajı yapılmalıdır. Mesafe uzunluğuna göre cihaz modeli seçilmelidir.

Görsel 5.30'da kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki beam dedektörlerinin görselleri görülmektedir.



Görsel 5.30: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz beam dedektörleri

Görsel 5.31'de beam dedektörlerinin hırsız alarm santraline dirençsiz ve dirençli bağlantı şemaları görülmektedir. Beam dedektörleri alıcı ve verici olmak üzere iki kısımdan meydana gelmektedir. Verici kısımda sadece besleme gerilimi bağlantı klemensleri, alıcı kısımda ise besleme gerilimi ve kontak uçları bağlantı klemensleri bulunur. Diğer dedektör bağlantılarında olduğu gibi kapalı kontak (NC-C) arasına dirençli ve dirençsiz bağlantı yapısına göre **zone** bağlantısı yapılır. Beam dedektörlerde verici ve alıcı kutularındaki lensler yardımı ile kızılötesi sinyaller uzak mesafelere iletilmekte ve sinyallerin birbirini görmesi sağlanmaktadır. Beam dedektörlerin kapakları kapalı olmaz ise dedektörler çalışmaz.



Görsel 5.31: Hırsız alarm santralinin zone uçlarına beam dedektörlerin bağlantısı

5.2.8. Hırsız Alarm Sistemlerinde Yangın Dedektörlerinin Kullanımı

Hırsız alarm sistemlerinde, yangın alarm sistemlerinde kullanılan dedektörler (duman, ısı, multi (duman+ısı), alev ve doğal gaz) ve yangın ihbar butonları kullanılabilir. Görsel 5.32'de yangın alarm sistemlerinde kullanılan farklı marka ve modellerdeki dedektörlerin ve yangın ihbar butonlarının görselleri görülmektedir.



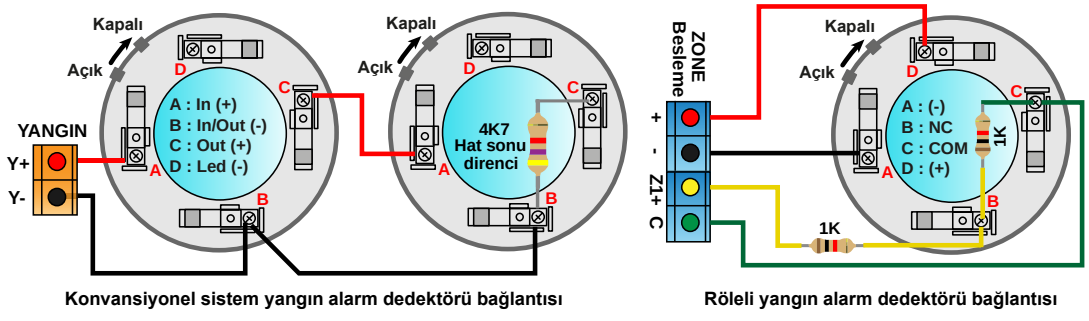
Görsel 5.32: Hırsız alarm sistemlerinde kullanılan yangın alarmı sensör çeşitleri

Hırsız alarm santrallerine yangın alarm sisteminde kullanılan ürünler iki şekilde bağlanabilir:

a) Konvansiyonel Sistem Yangın Alarm Dedektörü: Bu dedektörlerin hırsız alarm santrallerinde kullanılabilmesi için aynı konvansiyonel yangın alarm santrallerinde olduğu gibi konvansiyonel **zone** girişlerinin olması gerekir. Hırsız alarm santrallerinde, konvansiyonel yangın alarm dedektörlerinin bağlantısının yapılabilmesi için genellikle **Y+** ve **Y-** isminde veya **1. Zone** girişleri şeklinde klemensler bulunur.

b) Röleli Yangın Alarm Dedektörü: Röle çıkışlı bütün yangın alarm dedektörlerinin kapalı kontağı, (NC-C) hırsız alarm sistemindeki dedektörler gibi bağlantısı yapılarak kullanılabilir.

Yangın alarm sistemlerinde kullanılan ürünlerin hırsız alarm santraline bağlantısı Görsel 5.33'te görülmektedir.



Görsel 5.33: Hırsız alarm santraline yangın alarm ürünlerinin bağlantısı

5.2.9. Manyetik Kontak

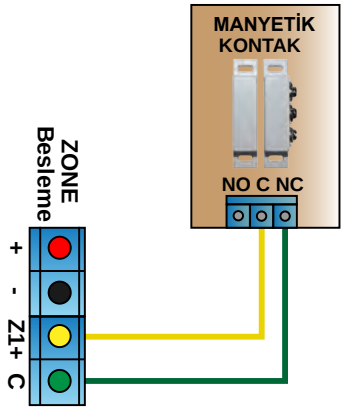
Hırsız alarm sistemlerinde en çok kullanılan ürünlerden bir tanesidir. Kapının veya pencerenin açılmasını algılayan cihazdır. Manyetik kontak cihazı, açılabilir hareketli noktalara monte edilir. Manyetik kontaklar iki kısımdan oluşur. Birinci kısmın içinde mıknatıs, ikinci kısmın içinde reed röle bulunur. Birinci kısım (mıknatıs) ikinci kısma (reed röle) yaklaştırıldığında mıknatıslanma gerçekleştiği için her iki kontak da hareket eder. Manyetik kontağın önüne mıknatıs yaklaştırıldığı zaman türüne göre açık ise kapalı, kapalı ise açık konuma geçerek görevini gerçekleştirir. Görsel 5.34'te kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı marka ve modellerdeki manyetik kontakların görselleri görülmektedir.



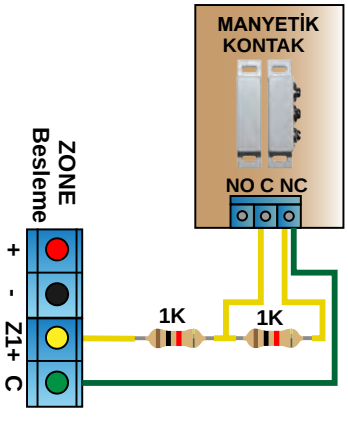
Görsel 5.34: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz manyetik kontaklar

Manyetik kontak, üzerinde kablo girişinin yapıldığı reed röle ve karşılığında mıknatıs olmak üzere iki parçadan oluşur. Kablo girişinin bulunduğu kısım kapı veya pencerenin kasasının yani sabit kalan kısmın üzerine montajlanır. Mıknatıs kısmı ise kapı veya pencerenin hareketli kısmı üzerine montajlanır. Sabit ve hareketli kısım arasındaki mesafe 1,5 cm- 3 cm arasında olmalıdır.

Görsel 5.35a'da manyetik kontakların hırsız alarm santralinin **zone** uçlarına dirençsiz, Görsel 5.35b'de ise dirençli bağlantı şemaları görülmektedir. **Zone** bağlantıları, dedektör çıkışlarının bağlantısı veya beam dedektör alıcı kısmı çıkışının bağlantısı ile aynıdır. Tek fark, manyetik kontak, mıknatıslanma prensibine göre çalıştığı için besleme gerilimi bağlantısının olmamasıdır.



Görsel 5.35a: Hırsız alarm santralinin zone uçlarına manyetik kontakların dirençsiz bağlantısı



Görsel 5.35b: Hırsız alarm santralinin zone uçlarına manyetik kontakların dirençli bağlantısı

5.2.10. Siren

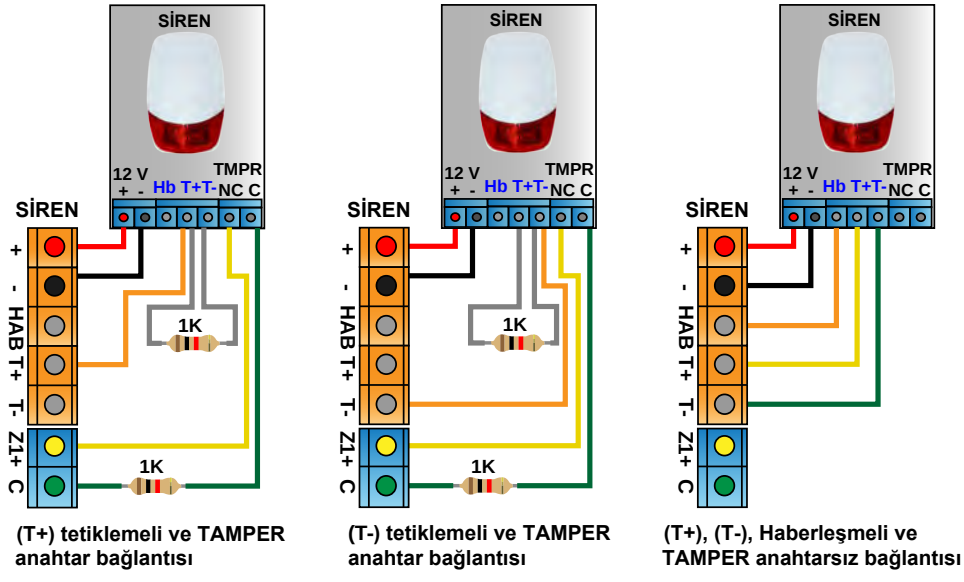
Hırsız alarm santraline dedektörlerden veya manyetik kontaklardan "Hırsızlık var." bilgisi geldiğinde sirenler aktif hâle gelir. Siren, hırsız alarm santralinden gelen tetikleme sinyaline göre (artı tetik T+, eksi tetik T-) sesli ve ışıklı hırsızlık olayı alarm bilgisi vererek insanları uyarır. Sirenlerin flaşörlü (ışıklı) ve flaşörsüz (ışıksız) çeşitleri vardır. Aynı zamanda iç mekânlarda kullanılmak üzere hazırlanan sirenlere **dâhilî siren**, dış mekânlarda kullanılmak üzere hazırlanan sirenlere de **haricî siren** denir. Dâhilî siren, genellikle 108-110 dB ses şiddetine sahiptir ve tavan yüksekliğinden 30 cm aşağıya montajı yapılabilir. Haricî siren ise genellikle 128 dB ses şiddetine sahiptir. Kötü niyetli kişilerin ilk yaptığı, dışarıda ses veren cihaza müdahale etmek olacağından cihazın

kolay ulaşamayacakları bir yüksekliğe montajının yapılması gerekir. Sirenin duvardan sökülmesi ve kapağının açılması durumunda alarm oluşabilmesi için sabotaj bağlantısının mutlaka yapılması gerekmektedir. Görsel 5.36'da yangın alarm sistemlerinde kullanılan farklı modellerdeki siren görselleri görülmektedir.



Görsel 5.36: Hırsız alarm sistemlerinde kullanılan dâhilî ve haricî sirenler

Görsel 5.37'de tetikleme türüne göre siren bağlantıları görülmektedir. Kullanılan modelin bağlantıları değişiklik göstereceğinden ürünün katalog bilgilerine mutlaka bakılmalıdır.



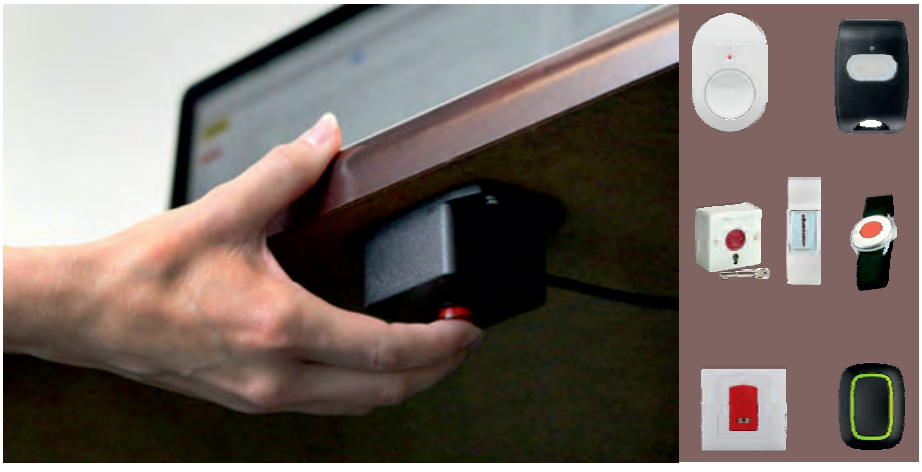
Görsel 5.37: Hırsız alarm sistemlerinde kullanılan siren bağlantıları

5.2.11. Hırsız Alarm Sistemlerinde Kullanılan Buton Çeşitleri

Hırsız alarm sistemlerinde kullanılan buton çeşitleri kuru kontak özelliğinde çalışır. Çalışma prensiplerinin mantığı NC veya NO olabilir. NC, normalde kapalı devre olan kontak yapısının butona basınca açık devre kontak yapısına dönmesidir. NO, normalde açık devre olan kontak yapısının butona basınca kapalı devre kontak yapısına dönmesidir.

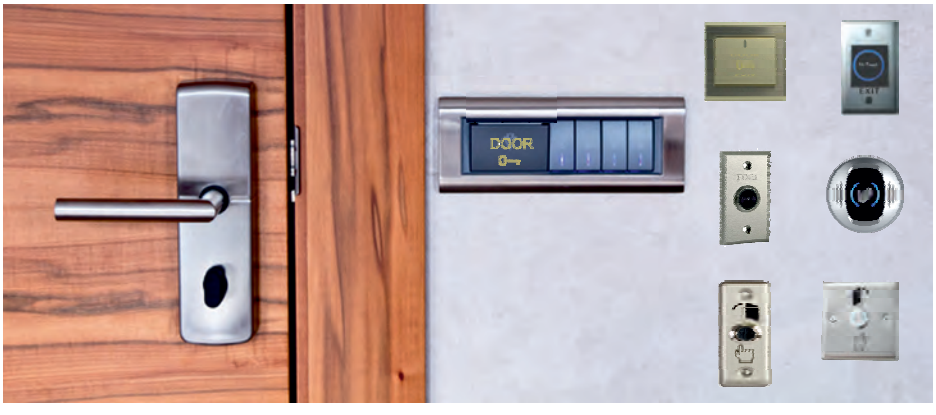
Hırsız alarm sistemlerinde kullanılan buton çeşitleri aşağıdaki gibidir:

1. Panik İhbar Butonu: NC kuru kontak mantığında çalışır. Yüksek güvenlik gerektiren yerlerde (banka, kuyumcu, döviz bürosu, adliye, belediye ve kamu binaları vb.) montajı duvar yüzeyine ve masaların alt tarafına yapılmalıdır. Alarm sistemi kurulu olsun veya olmasın devrede olmalıdır. 24 saat **zone** tipi olarak seçilmelidir ve **zone** tipi mutlaka sessiz **zone** olarak programlanmalıdır. Görsel 5.38'de kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki panik ihbar butonlarının görselleri görülmektedir.



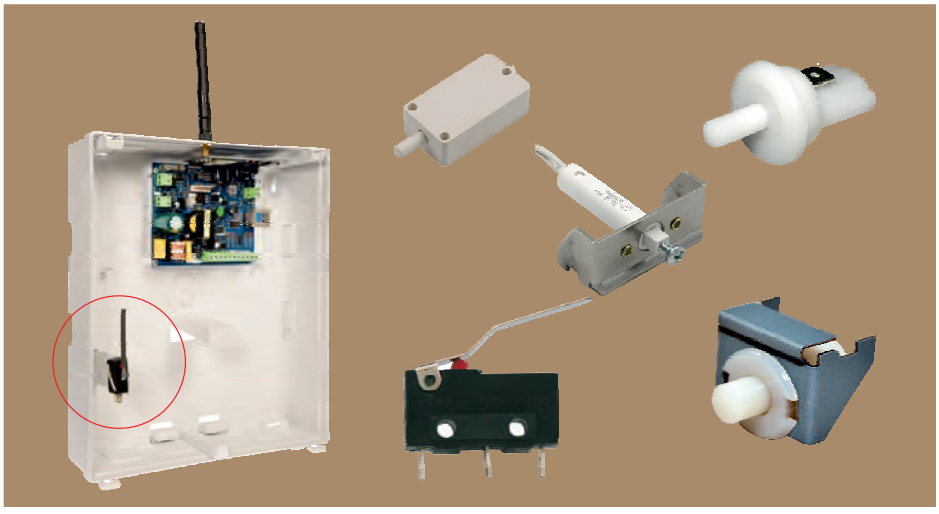
Görsel 5.38: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz panik ihbar butonları

2. Kapı Açma Butonu: NO kuru kontak mantığında çalışır. Montajı kapı girişlerinin yakınına yapılmalıdır. Yerden yüksekliği minimum 120 cm olmalıdır. Görsel 5.39'da kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki kapı açma butonlarının görselleri görülmektedir.



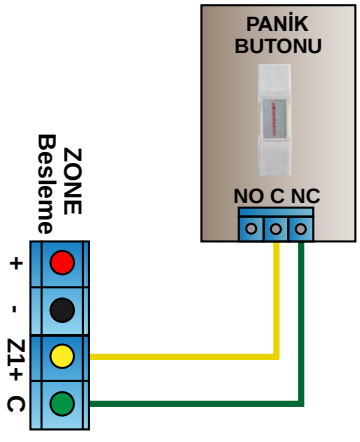
Görsel 5.39: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz kapı açma butonları

3. TAMPER Switch: NO kuru kontak mantığında çalışır. **TAMPER switch**, metal kutularda kapak açılmaları ve sökülmelerine karşı koruma sağlamak için kullanılır. Yani montajı alarm santrallerinin kapağına yapılabilir. Tüm hırsız alarm sisteminde kullanılan ürünlerin üzerinde bulunur. Dışarıdan kapakların açılması veya duvardan sökülmesine karşı koruma sağlamak amacıyla tasarlanmıştır. 24 saat **zone** tipi olarak programlanmalıdır. Hırsız alarm santrali kurulu olsun veya olmasın kapağı açıldığında veya duvardan söküldüğünde alarm vermelidir. Görsel 5.40'ta kablolu ve kablosuz hırsız alarm sistemlerinde kullanılan farklı modellerdeki **TAMPER switch**lerin görselleri görülmektedir.

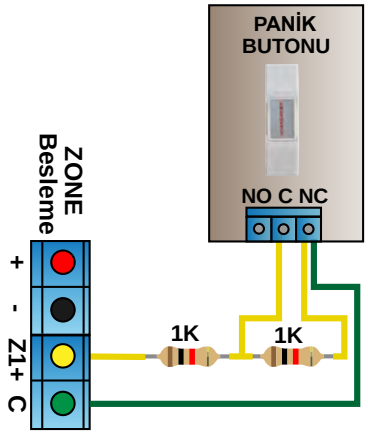


Görsel 5.40: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz TAMPER switchleri

Görsel 5.41a'da panik ihbar butonlarının hırsız alarm santralinin **zone** uçlarına dirençsiz, Görsel 5.41b'de ise dirençli bağlantı şemaları görülmektedir. Bağlantı şemalarında **zone** bağlantıları NC kuru kontak mantığında yapılmıştır.



Görsel 5.41a: Hırsız alarm santralinin zone uçlarına panik ihbar butonların dirençsiz bağlantısı



Görsel 5.41b: Hırsız alarm santralinin zone uçlarına panik ihbar butonların dirençli bağlantısı

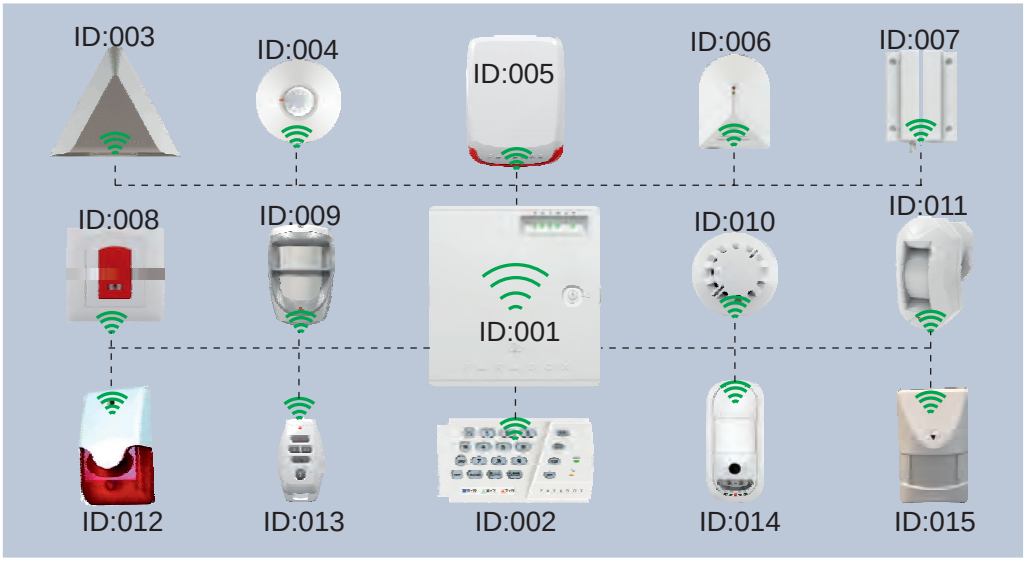
5.2.12. Kablosuz Hırsız Alarm Sistemlerinde Kullanılan Ürünler

Son zamanlarda teknolojinin ve iletişim sistemlerinin gelişmesi ile birlikte kablosuz hırsız alarm santrallerinin kullanımı artmaktadır. Kablosuz iletişimde genellikle **RF** ve **Wi-Fi** iletişim sistemleri kullanılmaktadır. 2010 yılından önce geliştirilen hırsız alarm sistemlerinde **RF** sinyallerinin iletişim frekansı 433 MHz iken 2010 yılından sonra geliştirilen hırsız alarm santrallerinde Avrupa Birliği tarafından 868 MHz olarak belirlenmiştir. Yine aynı zamanda **Wi-Fi** teknolojisinin gelişmesi ile birlikte hırsız alarm sistemi üreticileri **Wi-Fi** iletişim sisteminde ürünler geliştirmektedir.

Kablosuz hırsız alarm sistemlerinde kullanılan hırsız alarm santrali haricindeki ürünlerin besleme gerilimleri kalem pil ile sağlanmaktadır. Görsel 5.42'de kablosuz dedektörün hırsız alarm santraline bağlantı modeli görülmektedir. Kablosuz ürünler hırsız alarm santraline tanıtılırken iki yöntem kullanılır.

a) Seri Kod Numarası ile Tanıtma: Hırsız alarm sisteminde kullanılan ürünlerin içerisinde ürüne ait bir seri kod (ID) vardır. Bu kod **keypad** aracılığı ile kullanılan marka ve modelin programlama mantığına göre panele girilerek tanıtılır.

b) Anahtar ile Tanıtma: Hırsız alarm sisteminde kullanılan ürünlerin içerisinde bir **dip switch** anahtar-mikrobuton bulunur. Bu mikrobutona basılı tutularak ilgili ürünün hırsız alarm santrale tanıtımı yapılır.



Görsel 5.42: Kablosuz dedektörün hırsız alarm santraline bağlantı modeli

5.2.13. Alarm İzleme ve Kontrol

Hırsız alarm santrali üreticileri, ürettikleri sistemlerin uzaktan da izlenmesini sağlamak için ek üniteleri hırsız alarm santralinin elektronik kartına monte etmişlerdir. Bu sayede hırsız alarm sisteminin kurulumunun yapıldığı yerler uzaktan da kontrol altında tutulur. Hırsız alarm sistemini uzaktan izlemenin iki yöntemi vardır:

1. Haber Alma Merkezleri: Alarm Haber Alma Merkezi (AHM) kişilerin güvenliği için alarm izleme, kontrol ve haberleşme imkânları sunar. Görsel 5.43'te örnek bir haber alma merkezi uygulamasının görselleri görülmektedir. Haber alma merkezlerinin genel özellikleri şunlardır:

• Alarm anında siz ve belirlediğiniz kişiler aranarak bilgilendirme yapılır.

• Alarm türüne göre polis, itfaiye ve ambulans araması yapılarak eviniz veya iş yeriniz güvence altına alınır.

• Panik, yangın, hırsızlık, çevre güvenlik, hasta **(medikal)**, duman, ısı ve diğer alarm türlerini algılayarak acil durum **(duress)** alarmını alır ve doğrudan polis araması yapar.

• Günlük, haftalık ve aylık raporlama sağlanır. İnternet üzerinden günlük raporlamaları takip edilebilir.

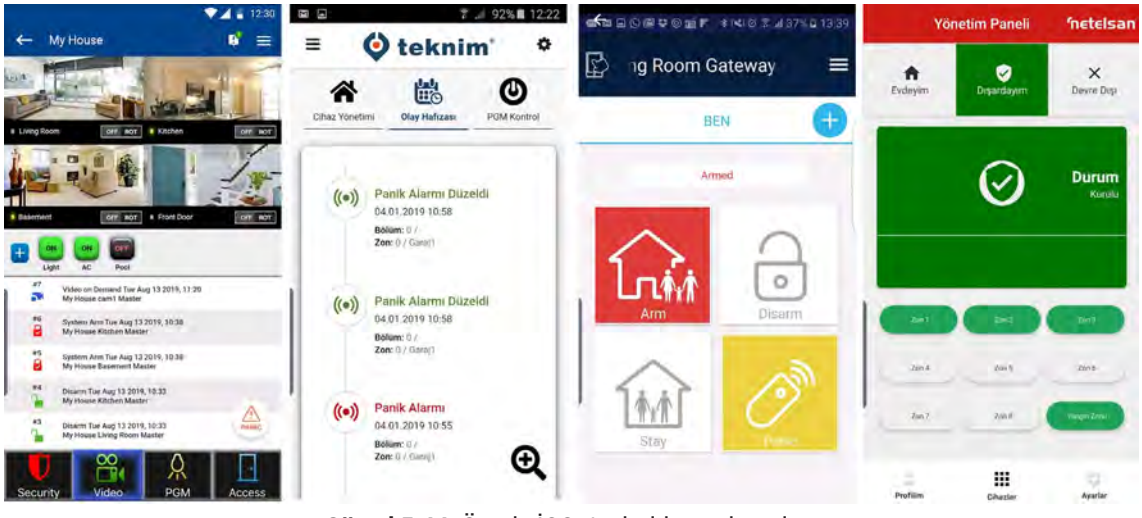
• Panel ile haberleşerek panel durumu sürekli kontrol edilir. Analog Hat (PSTN), internet ve GSM/GPRS üzerinden alarm bilgisi alınabilir.

• Panel ile ilgili güncellemeler otomatik yapılır.



Görsel 5.43: Örnek bir haber alma merkezi uygulaması

2. İOS-Android Uygulama Yazılımı: Teknolojinin gelişmesi ile birlikte hırsız alarm sistemleri üreticileri, geliştirdikleri İOS-Android uygulama yazılımları ile aynı haber alma merkezi gibi alarm izleme, kontrol ve haberleşme imkânları sunmaktadır. Görsel 5.44'te örnek IOS- Android telefonlar ile sistemin uzaktan kontrol uygulamasının görselleri görülmektedir.



Görsel 5.44: Örnek İOS-Android uygulamaları

İOS-Android uygulama yazılımlarının genel özellikleri şunlardır:

• İOS-Android uygulama yazılımları telefondan anlık bildirim gönderebilmektedir. Bu sayede telefondan alarmlar kontrol edilebilir, ayarlanabilir ve anlık bildirimler alınabilir.

• Cep telefonundan 24 saat boyunca hem iş yeri ve ev hem de herhangi bir yaşam alanına giren bir kişiden haber alınması sağlanır.

• Alarm türüne göre polis, itfaiye ve ambulans araması yapılarak ev veya iş yeri güvence altına alınır.

• İOS-Android uygulama yazılımları sayesinde hırsız alarm santrali ile haberleşme sağlanarak santralin durumu sürekli kontrol edilir. İnternet üzerinden günlük raporlamalar takip edilebilir. Hırsız alarm santralinin programları ile ilgili güncellemeler ya da değişiklikler yapılabilir.

5.2.14. Hırsız Alarm Sistemlerinde Kullanılan Yardımcı Ürünler

Hırsız alarm sistemlerinde aşağıdaki yardımcı ürünler kullanılmaktadır:

5.2.14.1. Uzaktan Kumanda Modülü

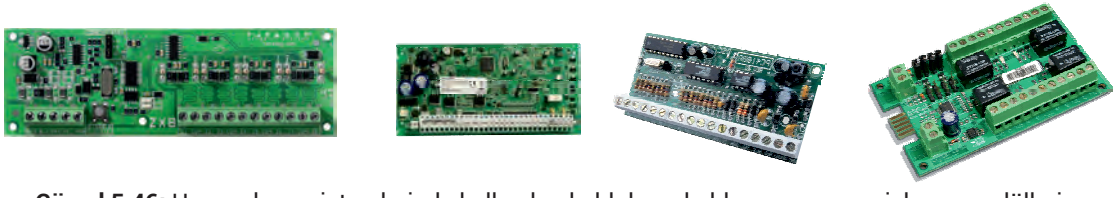
Uzaktan kumandalar, alarm sistemini aktif ve deaktif etmenin yanı sıra alarm sisteminin özelliklerine göre **zone** ekleme, çıkarma gibi farklı görevlerde de kullanılabilmektedir. Kumandanın ve alarm sisteminin özelliğine göre kumanda cihazı, en basit olarak alarm sistemini aktif ya da deaktif edip edemediği denenerek test edilebilmektedir. Görsel 5.45'te hırsız alarm sistemlerinde kullanılan uzaktan kumanda görselleri görülmektedir.



Görsel 5.45: Hırsız alarm sistemlerinde kullanılan uzaktan kumandalar

5.2.14.2. Kablolu-Kablosuz Zone Genişleme Modülleri

Kablolu-kablosuz **zone** genişleme modülleri, mevcut alarm panelinde bulunan **zone** sayısını arttırmak için kullanılır. 4, 8, 16. **zone** gibi **zone** genişleme kartları vardır. Kullanılan marka ve modeldeki hırsız alarm santraline uygun **zone** genişleme modülleri tercih edilmelidir. Genişleme modülleri **zone** girişini anti-**TAMPER switch** (sabotaj) girişi olarak kullanma ve otomatik olarak panel tanıma özelliğine sahiptir. Kablosuz **zone** genişleme modülleri RF 868 MHz veya **Wi-Fi** olarak da çalışabilir. Kablosuz **zone** genişleme modülleri 32 adet **zone**a kadar desteklemektedir. Görsel 5.46'da hırsız alarm sistemlerinde kullanılan kablolu-kablosuz **zone** genişleme modüllerinin görselleri görülmektedir.

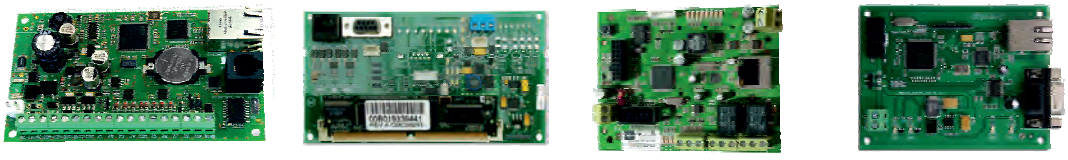


Görsel 5.46: Hırsız alarm sistemlerinde kullanılan kablolu ve kablosuz zone genişleme modülleri

5.2.14.3. LAN (Network) Modülü – İnternet İletişim Modülü

LAN (Network Modülü) ile hırsız alarm sistemlerine uzaktan erişim sağlanır. Network modülü ile bir sisteme bağlanma aplikasyon uygulaması, PC yazılımı ile mümkündür. Programlama, yazılım yükseltme ve izlemenin yanı sıra raporlama yapabilir. Alarm izleme merkezi (haber alma merkezi) alıcılarına internet bağlantı yoluyla raporlama yapar. Ayrıca internet bağlantısı olmadan kapalı network ağı olarak çalışacak şekilde yapılandırılabilir. İnternet modülü, uzaktan erişim sağlayan iki çıkış içerir. Web arayüzü veya aplikasyon uygulaması aracılığıyla yapılandırılır. Işıkları, ısıtıcıları ve benzerlerini kontrol etmek için kullanılabilir.

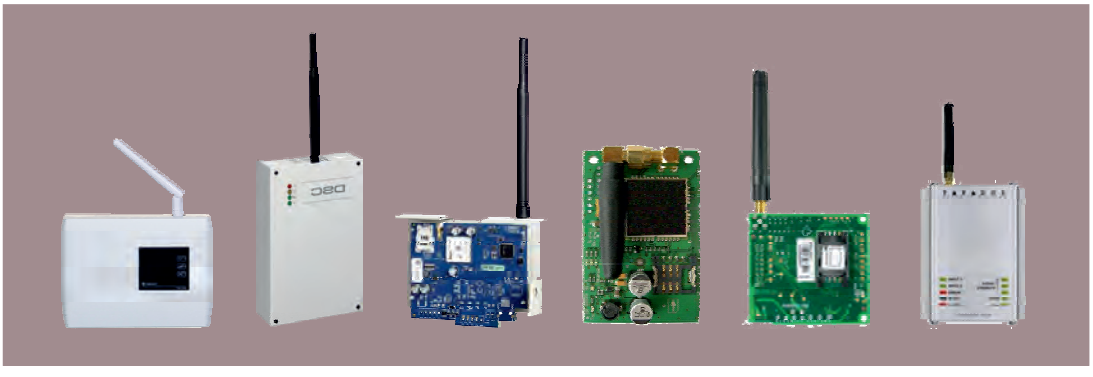
Görsel 5.47'de hırsız alarm sistemlerinde kullanılan network modüllerinin görselleri görülmektedir.



Görsel 5.47: Hırsız alarm sistemlerinde kullanılan network modülleri

5.2.14.4. GSM/GPRS Haberleşme Modülleri

GSM/GPRS haberleşme modülleri ile hırsız alarm santrallerine uzaktan erişim sağlanır. Haberleşme modülleri ile mobil cihazlar aracılığıyla bağlantı sağlama (aplikasyon uygulaması), merkezi izleme istasyonuna raporlama, SMS gönderme ve kişisel sesli arama, 4G/3G/2G veri iletişimi veya GSM sesli raporlama çözümleri, 2G veri hızı ve/veya GSM sesi sağlarken raporlama imkânları da sunar. Hücresel modüller ile güvenilir raporlama ve kontrol panelleri arasında iki yönlü iletişim, mobil uygulama ve CMS'nin (haber alma merkezi yazılımı) yanı sıra uzak site üretici yazılımı yükseltmeleri de yapılabilir. Görsel 5.48'de hırsız alarm sistemlerinde kullanılan GSM/GPRS modüllerinin görselleri görülmektedir.



Görsel 5.48: Hırsız alarm sistemlerinde kullanılan GSM/GPRS modülleri

5.3. HIRSIZ ALARM SİSTEMİNİN PROGRAMLANMASI

Hırsız alarm sistemlerinde santral ile kullanılan diğer ürünlerin montajı, bağlantısı yapıldıktan ve yapılan bağlantıların kontrolü sağlandıktan sonra sistemin aktif edilmesi gerekir. Hırsız alarm sistemlerini aktif etmek için öncelikle hırsız alarm santrali ile **keypad** bağlantısının mutlaka yapılması gerekir. Daha sonra aşağıdaki işlemler sıra ile uygulanmalıdır:

• Hırsız alarm santralinin şehir şebekesi (AC 220 V) ile transformatör bağlantısı gerçekleştirilir. Akü bağlantıları yapılır.

• Hırsız alarm santralinin ve diğer kullanılan ürünlerin enerji ışıkları kontrol edilir.

• Hırsız alarm sisteminde kullanılan ürünlerin enerji ışıkları normal ise sistemin çalışması için her bir **zone**nun tanıtılması gerekir. Tanıtımı yapıldıkça **zone**a bağlanan ürünlerin çalışması kontrol edilir.

• Hırsız alarm santralinin tarih ve saat ayarları yapılır.

• Örnek bir kullanıcı şifresi ile hırsız alarm santrali kurulup çalışması kontrol edilir. Kullanılan dedektörler algılama yapıyor, sirenler çalışıyor ise yine aynı kullanıcı şifresi ile hırsız alarm santrali kurulumu iptal edilip **standby** (bekleme) moduna alınır.

• **Zone** özelliklerine göre sürelerin programlanması (giriş, çıkış, siren çalma, PGM süresi vb.) gerçekleştirilir.

• Kullanıcı şifreleri tanımlanır. Sistem haber alma merkezi veya İOS-Android yazılımları ile entegre edilerek hırsız alarm sisteminin kurulumu tamamlanır.

Hırsız alarm sisteminde kullanılan ürünlerin montajı ve bağlantıları yapıldıktan sonra sistemin çalışabilmesi için programlama modunda **zone**ların tanıtımı, giriş ve çıkış süreleri, siren çalma süresi, kullanıcı şifreleri, haber alma merkezinin tanıtımı ve diğer işlemler yapılmalıdır.

NOT

Hırsız alarm santralinin programlama mantığı farklı marka ve modellere göre değişmez. Sadece işlem sırası veya işlem kodları farklı olabilir. Kullanılan marka ve modelin ürün katalogları ve program kılavuzları kontrol edilerek işlem yapılabilir.

5.3.1. Keypadin Tanıtılması

Hırsız alarm santralini programlayabilmek için **keypad** bağlantısının (kablolu-kablosuz) yapılması gerekir. Bütün programlama işlemleri **keypad** ile yapılır. Hata ve alarm durumları **keypad** aracılığı ile gözlemlenir.

Kablolu **keypad**ler hırsız alarm santraline tanıtılırken herhangi bir işleme gerek yoktur. Hırsız alarm santraline ve **keypad**e enerji verildiğinde **keypad** kablolu olarak bağlandığı için hırsız alarm santrali otomatik olarak **keypad**i tanır. Kablosuz **keypad** kullanılacak ise hırsız alarm santraline ve kablosuz **keypad**e enerji verildiğinde 15 dk. içerisinde **keypad** üzerinden tuş kombinasyonuna basılıp panel ile haberleşmesi sağlanarak tanıtım yapılabilir. Örneğin ON-OFF tuşu ve aydınlatma tuşuna basılı tutularak kablosuz **keypad** tanıtımı yapılabileceği gibi **keypad**in arka kapağındaki **switch**e basılıp çekildiğinde de hırsız alarm santrali ile **keypad** iletişime geçer ve otomatik tanıtımı yapılmış olur.

5.3.2. Fabrikasyon Şifreler

Hırsız alarm santrali üreticileri, santralin programlanabilmesi için aşağıdaki kodları sabit olarak tanımlayarak gönderirler. İsteğe bağlı olarak bunlar değiştirilebilir.

• **Mühendislik Kodu- Montajcı Kodu:** Mühendislik kodu ile hırsız alarm sisteminin bütün ayarları ve programlamaları yapılabilir. Mühendislik kodu ile aynı zamanda şifre değişikliği hariç **master** kodu ile yapılan işlemler de gerçekleştirilebilir.

• **Master Kodu: Master** kodu ile hırsız alarm sistemine ait tarih, saat, telefon numaraları, kullanıcı şifreleri vb. ana kullanıcı ayarları menülerine erişim sağlanır. **Master** kod bazı marka ve modellerde **Kısım 1 ve Kısım 2** olarak ayrı ayrı verilebilir. Eğer sistemde kısımlandırma olmaz ise **Master Kod 2 Kısım 1** atanacaktır.

• **Bakımcı Kodu:** Montajcı kodu gibi olup belirli kısıtlamalara sahiptir. Bakımcı kodu ile programlama menüsüne giriş sağlanır. Programla ile ilgili tüm özellikleri programlamayı sağlar. Kullanıcı kodları ve panelin haberleşmesi ile ilgili menülere erişim sağlanamaz.

•**Tehdit (Duress) Kodu:** Herhangi bir tehdit durumunda kullanılan koddur. Alarmı kurma ve açma yapabilir. Kurma veya açma durumunda izleme merkezine **tehdit altında sistem kuruldu-açıldı** bilgisi gönderir. İzleme merkezi tehdit kodu sinyali düştüğünde olay yerine polis gönderir.

• **Kullanıcı Kodu:** Hırsız alarm santralinin kurulumu ve programlanması tamamlandıktan sonra sadece hırsız alarm sistemini kurma ve açma işlemi için kullanılır.

• **Panel Resetleme:** Hırsız alarm santralini fabrikasyon ayarlarına döndürmek için kullanılır. Panel resetleme işlemi, **keypad**den **ENTER>0000>950>ENTER** sırası ile yapılabildiği gibi şu şekillerde de yapılabilir:

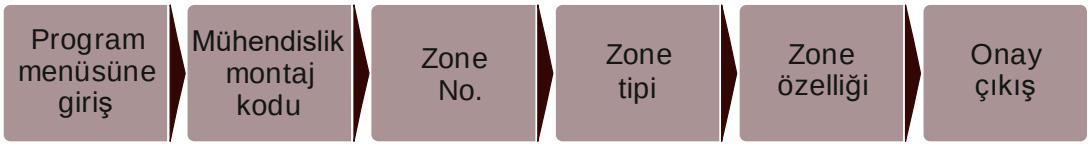
Hırsız alarm santrali üzerinde **Zone 1 i**le PGM klemensleri kısa devre yapılıp elektrik ve akü bağlantısı kesilir. **Keypad** normal çalışmaya döndüğü zaman kısa devre kaldırılır. Bu şekilde panel resetlenmiş olur.

Hırsız alarm santralinin kartı üzerindeki **reset** butonuna 5 sn. basılı tutulur. Kart üzerindeki LED'ler hızlı hızlı yanıp sönmeye başladığında el **reset** butonundan çekilip tekrar 5 sn. basılı tutularak da resetleme işlemi gerçekleştirilir.

5.3.3. Zone Programlanması

Hırsız alarm santrallerinde **zone** uçlarına dedektör ve manyetik kontak gibi ürünler bağlanır. Bu ürünler sayesinde hırsızlık olayının erken teşhis edilmesi sağlanır. Hırsız alarm santraline enerji verildiğinde **zone** uçları aktif değildir. Herhangi bir algılama yapamaz. Bunun için **zone** uçlarının **keypad** aracılığı ile programlanarak aktif edilmesi gerekir.

Görsel 5.49'da hırsız alarm santralinin **zone** uçlarının programlanmasının blog diyagramı görülmektedir. Kullanılan hırsız alarm santralinde **zone** programlanması yapılırken öncelikle ilgili tuş takımı+mühendislik kodu ile programlama menüsüne girilir. **Zone** numarası iki veya üç haneli olarak girilir. **Zone** tipi tanımlanır. Atanacağı kısım belirlenir. **Zone** özellikleri belirlenerek ONAY işleminden sonra **zone** tanıtılmış olur.



Görsel 5.49: Hırsız alarm santralinde zone programlanması blok diyagramı

5.3.3.1. Zone Program Menüsüne Girilmesi

Hırsız alarm santrallerinde **zone** tanımlayabilmek için önce programlama menüsüne girilmesi gerekir. Programlama menüsüne ilgili tuş takımı ve mühendislik (montajcı) kodu ile giriş yapılır. Bu işlemde genelde kullanılan modele göre farklılık gösterir. Farklı modele göre programlama menüsüne girilmesi örnekleri aşağıda verilmiştir:

Örnek : ENTER / 0000

5.3.3.2. Zone Numarasının Girilmesi

Hırsız alarm santrallerinde kaç numaralı **zone** programlanacak ise ilgili **zone** numarası kullanılan modele göre iki veya üç hane olarak girilir.

Örnek 1: 01

Örnek 2: 001

5.3.3.3. Zone Tipinin Belirlenmesi

Hırsız alarm santrallerinde **zone** tipinin belirlenmesi çok önemlidir. Çünkü **zone** tipine göre ilgili **zone**nun çalışması belirlenir. **Zone** tipi belirlenirken **keypad**den ilgili **zone** tipine karşılık gelen iki haneli kod girilir. Bu iki haneli kod kullanılan modele göre farklılık gösterir. Aynı işlevi yerine getiren **zone** tipi bir markada 03 ile gösterilirken farklı bir modelde 05 ile gösterilebilir.

Örnek 1: 03

Örnek 2: 05

Genellikle kullanılan zone tiplerinin işlevleri aşağıdaki gibidir:

• **Kapalı Zone Tipi:** Kullanılmayan **zone**lardır. Gelen bilgiler hiçbir şekilde değerlendirilmez.

• **Uyarı Zone Tipi:** Gelen bilgi hırsız alarm sistemi kurulu olsa da olmasa da **keypad**de görsel olarak görülür ve **buzzer**dan **1 sn.lik** kısa bir ses uyarısı verilir.

• **24 Saat Zone Tipi:** Su kaçağı, duman dedektörü, gaz dedektörü, panik butonu, cam kırılma dedektörü, darbe dedektörü, siren sabotajı vb. acil durum sensörlerinin bağlantısında kullanılabilecek, hırsız alarm sistemi kurulu olsa da olmasa da sistemi alarm moduna geçirecek **zone** tipidir.

• **Ani Zone Tipi:** Hırsız alarm sistemi kurulu iken alarm bilgisi alındığı anda giriş gecikme süresini beklemeden sistemi alarma geçiren, siren ve flaşörden uyarı veren **zone** tipidir.

• **Normal (Giriş gecikmeli) Zone Tipi:** Hırsız alarm sistemi kurulu iken alarm bilgisi geldiğinde giriş gecikme süresini başlatan ve bu süre zarfında şifre girilmezse sistemi alarma geçiren **zone** tipidir.

• **Sessiz Zone Tipi:** Alarm bilgisi geldiğinde **keypad**de sadece görsel olarak ikaz veren, hırsız alarm santralini alarm moduna geçirmeyen **zone** tipidir. Sessiz **zone**da bir alarm oluşursa hırsız alarm santrali merkezî istasyona sireni tetiklemeden alarm durumunu rapor edecektir.

NOT

Eğer **zone** 24 saat yangın **zone**u olarak tanımlanmış ise hırsız alarm santrali sessiz **zone** tanımına rağmen alarm modunda sesli bir alarm verecektir.

• **Bypass Zone Tipi:** Bypass özelliği, geçici olarak belirlenen bölgelerden alarm verilmemesini sağlar. Örnek olarak misafir odası günlük kullanımda kuruluyorken misafir geldiğinde kurulmaması gereklidir. Bu durumda o bölge bypass yapılır ve o bölgede hırsız alarm sistemi devre dışı bırakılır. Tekrar kurulana kadar alarm durumu oluşmaz. Bypass işlemi bir defaya mahsustur. Alarm kurulup açıldığında bypass özelliği devre dışı kalır.

• **Chime (Zil) Zone Tipi:** Hırsız alarm sistemi devre dışı durumdayken zil **zone**u, özelliği verilen bölgelerden herhangi bir hareket algılandığında uyarı veya melodi sesi verir. Bu özellik işletmeler ve evler için idealdir. İşletmelerde çalışanlar, giriş kapısından gelen biri olursa içerde çalışırken uyarı sesini duyup birinin geldiğini anlayabilir. Evlerde bu uygulama giriş kapısı ve çocuk odası kontrolleri için uygulanmaktadır.

• **Creep Zone Tipi:** PIR dedektörlerde kullanılan bu önemli güvenlik özelliği PIR dedektörlerin alt kısmının algılanmasını sağlar ve dedektörlerin alt kısmında geçişleri engelleyerek güvenlik açığı oluşturmadan sağlıklı bir koruma yapar.

• **Force Zone Tipi: Zone** açık iken kurulmayı sağlar. Giriş gecikmeli **zone** haricinde kullanımı önerilmez.

• **Takip Zone Tipi:** Takip **zone**u giriş gecikmeli bölgeyi takip eden **zone** anlamındadır. Giriş kapısına manyetik kontak bağlı olduğu düşünüldüğünde giriş kapısı açıldıktan sonra PIR dedektörü

algılama yaparsa giriş süresi kadar süre vermeye devam eder. Ancak kişi giriş kapısından değil de camdan içeri girerse o zaman herhangi bir süre tanımlamadan ani olarak çalışıp alarm oluşturur. Takip **zone**nuna örnek olarak giriş kapısında manyetik kontak ve içeride giriş kapısını gören PIR dedektörü olsun. Giriş kapısı açıldıktan sonra PIR dedektörü içeri giren kişiyi görünce giriş gecikmesi kadar süre verir. Ancak kişi kapıdan değil de camdan içeri girerse süre vermez ve ani olarak alarm verir. **Zone** tanımlamalarından hangi **zone**ların takip **zone**u olacağı seçilmelidir.

• **Akıllı Zone Tipi:** Akıllı **zone** özelliği verilen PIR dedektörü, gördüğü ilk hareketi hafızasına alır ve akıllı **zone** süresi boyunca beklemede kalır. Tekrar o zaman içerisinde bir hareket daha görürse alarm bilgisi oluşturur. Akıllı **zone**un alarm oluşturabilmesi için girilmiş olan zaman aralığında üst üste iki defa hareket algılaması gerekir. Akıllı **zone**ların bu özelliği sayesinde yanlış alarmların önüne geçilir.

5.3.3.4. Zone Kısımlandırılması

Hırsız alarm santralleri programlanırken kendi içinde kısımlara ayrılabilir. Buradaki amaç hırsız alarm santralinin **zone**larına bağlı olan ürünleri bölümlere ayırabilmektir. Ayrılan her kısmın ayrı bir şekilde kurulumu ve tanıtımı yapılabilir. Genellikle hırsız alarm santrallerinde **Kısım 1 ve Kısım 2** olarak tanımlanan, birbirinden bağımsız iki alana bölünebilecek kısımlandırma özelliği vardır. Kısımlandırma, paylaşımlı güvenlik sistemlerinin daha pratik olduğu ev, ofis veya depo binalarındaki montajlarda kullanılabilir. Kısımlandırıldığında her **zone**, her kullanıcı kodu ve sistemin birçok özelliği **Kısım 1, Kısım 2** veya her iki kısma birden atanabilir.

NOT Eğer sistem kısımlandırılmamışsa tüm **zone**lar, kullanıcı kodları ve özellikler **Kısım 1**'e aitmiş gibi kabul edilir.

Örnek 1: 1 : 1. Kısma atama yapıldı.

Örnek 2: 2 : 2. Kısma atama yapıldı.

5.3.3.5. Zone Özellikleri

Hırsız alarm santralinde ilgili **zone**nun kısım atamasından sonra **zone** özellikleri veya opsiyonları **keypad**de 1 ile 8 arasındaki tuşların üzerindeki ışıkların yanık-sönük durumuna göre ayarlanabilir. **Keypad**deki bu 1-8 arası tuşlarının üzerindeki ışıkların anlamları aşağıdaki gibidir:

[1] : Otomatik **zone** kapanması [2] : Baypass edilebilir **zone** [3] : RF **zone** denetimi
[6] : Akıllı **zone** [7] : Gecikmeli alarm aktarımı [8] : Force **zone**

[4] : KAPALI	[5] : KAPALI	Sesli Alarm	[4] : KAPALI	[5] : AÇIK	Darbeli Alarm
[**4**] : AÇIK	[5] : KAPALI	Sessiz Alarm	[**4**] : AÇIK	[5] : AÇIK	Sadece Rapor

Örnek 1: [**1**] [**2**] [3] [4] [5] [6] [7] [8] : 1, 2 ve 8. özellikler ilgili **zone**a eklendi.

Örnek 2: [**1**] [**2**] [3] [4] [5] [6] [7] [8] : 1, 2, 3 ve 8. özellikler **zone**a eklendi.

5.3.3.6. Zone Onayı ve Zone Programından Çıkış

Hırsız alarm santralinde **keypad** aracılığı ile programlama menüsüne girilip **zone** numarası, **zone** özellikleri, kısım ataması ve 1-8 arası seçeneklerden opsiyonlar seçildikten sonra kullanılan marka ve modele göre ilgili tuş takımına basılarak **zone** tanımlama işlemi tamamlanır. Hırsız alarm santrali programlama moduna tekrar döner. İkinci **zone** numarası, **zone** özellikleri ve kısım ataması yapılarak **zone** programlamasına devam edilebilir.

Zone programdan çıkış tuş kombinasyonuna basılarak hırsız alarm santrali programlama modundan çıkarılır.

Örnek 1: ENTER: **Zone** tanıtımı tamamlandı. Diğer **zone** programlamasına devam edilir.
Örnek 2: ENTER / CLEAR: **Zone** tanımlaması yapıldı. Programlama modundan çıkıldı.

5.3.3.7. Zone Programlama Örnekleri

Hırsız alarm santralindeki **zone** programlamasına örnekler Tablo 5.1'de verilmiştir. 1. örnekte 1 numaralı **zone**, ani **zone** tipinde, 1. Kısma, 1, 2 ve 8 numaralı özellikleri aktif olarak tanımlanmıştır. 2. örnekte 2 numaralı **zone**, 24 saat **zone** tipinde, 2. Kısma, 1, 2 ve 8 numaralı özellikleri aktif olarak tanımlanmıştır.

Tablo 5.1'deki verilen örnekler kullanılan hırsız alarm santralinin modeline göre farklılık gösterebilir.

Tablo 5.1: Hırsız Alarm Santralinde Zone Programlama Örnekleri

Örnek	Programlama Menüsüne Giriş	Mühendislik Kodu	**Zone** Numarası (3 Haneli	**Zone** Tipi (2 Haneli)	Kısım Ataması (1, 2, 3)	**Zone** Özellikleri (1-8)	Onay / Çıkış
1	ENTER	0000	001	(2 Haneli)	1	1,2,3,4,5,6,7,8	Enter/Clear
2	ENTER	0000	002	16	2	1,2,3,4,5,6,7,8	Enter/Clear

5.3.4. Tarih ve Saat Programlama

Hırsız alarm santralinin tarih ve saat ayarı mutlaka yapılmalıdır. Çünkü haber alma merkezi veya İOS-Android uygulamaların tam zamanlı olarak hırsız alarm santrali ile haberleşmesinde önemlidir. Hatta bazı hırsız alarm santralleri tarih ve saat programını girmeden diğer programların yapılmasına müsaade etmemektedir. Bazı hırsız alarm santrallerinde ise tarih ve saat ayarı girilmeden bütün programlama yapılabilmektedir. Kullanıcı şifresi ile sistem kurulup açılabilir. Sadece hırsız alarm santralini otomatik kurma özelliği aktif olmamaktadır.

Hırsız alarm santralinin tarih ve saat programları yapılırken kullanılan modelin ilgili tuş takımı + **master** kodu, ardından önce saat (17) ve sonra dakika (30) bilgisi 2 haneli olarak (12 saat sistemi için 1, 24 saat sistemi için 2) girilir. Önce yıl (2022) bilgisi 4 haneli olarak, sonra ay (12) bilgisi 2 haneli olarak daha sonra da gün (19) bilgisi 2 haneli olarak girilip onay ve çıkış işlemi yapılır.

Örnek : [İlgili Tuş Takımı] / [**MASTER** KODU] / [TBL] / [5] / [SA:DA] / [1] / [YYYY/AY/GG] > [CLEAR]

5.3.5. Sürelerin Programlanması

Hırsız alarm santrallerinde **zone** tanımlamalarından sonra **zone**un giriş süresi, çıkış süresi, sirenin çıkış süresi ve otomatik kurulum zamanının istenen süreler içinde ayarlaması yapılabilir. Hırsız alarm santrallerinde fabrikasyon olarak şifreler ve zamanlamalar mevcuttur. Herhangi bir programlama yapılmasa dahi hırsız alarm santrali ilgili şifreler ile kurulup çözülebilir.

Hırsız alarm santralinde süreleri programlamak için önce programlama menüsüne mühendislik kodu ile giriş yapılır. Ardından ayarı yapılacak sürenin giriş kodu girilir. Ayarı yapılacak giriş ve çıkış gecikme süresinin zaman ayarı saniye cinsinden, siren çıkış süresinin zaman ayarı dakika

cinsinden 3 haneli olarak girilip onay ve çıkış işlemi yapılarak gerçekleştirilir. Giriş ve çıkış gecikmesi maksimum 255 saniye, çıkış gecikme süresi maksimum 255 dakikadır.

Giriş gecikmesi, hırsız alarm santrali kurulu iken dedektörlerden gelen hırsız sinyalinin algılanması ve hırsız alarm santralinin alarm moduna geçmeden önce **keypad**den kullanıcı şifresi girilip alarm modu pasif edilerek kurulu olan sistemin açılması için gereken süredir. Örneğin okulda atölye binasında hırsız alarm sistemi kurulu iken bina ana kapısı açıldığında **keypad**e kullanıcı şifresi ile sistemi açık duruma getirmek için verilen süredir.

Örnek : ENTER / 0000 / 710 / 030 / ENTER / CLEAR: Giriş gecikmesi 30 sn. olarak belirlendi.

Çıkış gecikmesi, hırsız alarm santrali kullanıcı şifresi ile kurulduğunda ilgili fiziki mekândan alarm verdirmeden çıkmak için kullanılır. Örneğin okulda atölye binası akşam boşaltılırken hırsız alarm sisteminde **keypad**e kullanıcı şifresi ile girip, sistemin kurulumunu yapıp atölye kapısını hırsız alarm santrali kurulmadan kilitleyip çıkmak için gereken süredir.

Örnek : ENTER / 0000 / 745 / 020 / ENTER / CLEAR: Çıkış gecikmesi 20 sn .olarak belirlendi.

Siren çıkış süresi, siren çalma süresidir. Özelliği, alarm anında belirlenen zaman kadar çalışıp durur. Tekrar alarm olursa yine belirlenen zaman kadar çalışıp durur. Bu durum, sistem devre dışı bırakılana kadar tekrar edilir.

Örnek : ENTER / 0000 / 747 / 010 / ENTER / CLEAR: Siren çıkış süresi 10 dk. olarak belirlendi.

5.3.6. PGM Programlama

Hırsız alarm santrallerinde PGM çıkışları programlanabilir çıkışlardır. PGM çıkışları programlanırken bir olay ve olayın sonucunda ne kadar zaman aktif olacağı belirlenir. Örneğin, **Zone 1**'de alarm durumu oluştuğunda aktif hâle gelsin ve 060 sn. sonra normale dönsün biçiminde programlanabileceği gibi **Zone 1** normale döndüğünde devre dışı olsun veya şifre girilince normale dönsün biçiminde de programlanabilir.

Hırsız alarm santralinde PGM programlamak için önce programlama menüsüne mühendislik kodu ile girilir. Sonra sırayla PGM aktivasyonu veya aktivasyon iptalinin giriş kodu, kısım numarası ve **zone** numarası girilir. Herhangi bir kullanıcı numarası girilerek PGM programlaması yapılabilir.

Örnek 1: [220] / [01] / [02] / [01] / [ENTER] / [CLEAR]: 2 numaralı **zone** açıldığında PGM aktif yapıldı.

Örnek 2: [221] / [01] / [02] / [01] / [ENTER] / [CLEAR]: 2 numaralı **zone** kapandığı zaman PGM devre dışı olur.

PGM çıkış süresi için önce programlama menüsüne mühendislik kodu ile girilir. Ardından PGM çıkış süresinin giriş kodu, PGM çıkış süresinin zaman ayarı saniye cinsinden 3 haneli olarak girilip onay ve çıkış işlemi yapılır. Maksimum PGM çıkış süresi 255 saniyedir. Herhangi bir süre girilmese bile siren çıkış süresi boyunca aktif olur.

Örnek 3: [281] / [0 / 1 / 0] / [ENTER] / [CLEAR]: PGM çıkış süresi 10 saniye olarak belirlendi.

Örnek 4: [281] / [0 / 0 / 0] / [ENTER] / [CLEAR]: PGM çıkış süresi yok yani 0 saniye olarak belirlendi.

5.3.7. Zamanlı Otomatik Kurulumun Programlanması

Hırsız alarm santralinin otomatik kurulum yapılabilmesi için kullanılır. Gelişmiş hırsız alarm santrali modellerinde hafta içi, hafta sonu veya hafta içi farklı saatlerde hırsız alarm sisteminin otomatik olarak kurulması yapılabilir. Örneğin okulumuzda hafta içi her gün saat 18.00'den sonra otomatik kurulum gerçekleştirilebilir.

Zamanlı otomatik kurulum yapılırken önce programlama menüsüne mühendislik kodu ile girilir. Kısım zamanlı otomatik girme aktif edilir. İlgili kısma otomatik kurulum saati atanıp onay ve çıkış işlemi yapılarak gerçekleştirilir.

Örnek 1: [741] / [1] / [ENTER] / [CLEAR]: 1. Kısım zamanlı otomatik kurma seçeneği aktif edildi.

Örnek 2: [761] / [18:00] / [ENTER] / [CLEAR]: 1. Kısım otomatik kurma zamanı girilir.

5.3.8. Kablosuz Ürünlerin Programlanması

Kablosuz hırsız alarm santrallerinde, kablosuz dedektör, manyetik kontak ve PGM modüllerinin tanıtımı aynıdır. Kablosuz hırsız alarm santralinde kablosuz ürünleri tanıtmak için önce programlama menüsüne mühendislik kodu ile girilir. **Zone** numarası girildikten sonra kablosuz dedektörün içindeki 6 haneli seri numarası yazılıp onay ve çıkış işlemi yapılır. Programlamaya başlamadan önce ilk olarak eğer kablolu **zone** kullanıldıysa kablolu **zone**lar tanıtılmalıdır. Hırsız alarm santralinin anteni takılı olmalıdır. Ayrıca tanıtılmış bir **zone**u silmek için seri numarası kısmına 000000 yazılır.

Örnek 1: ENTER / [0000] / [061] / 6 haneli seri No. / [ENTER] / [CLEAR]: 1. **Zone**a kablosuz ürün tanıtımı

Örnek 2: ENTER / [0000] / [092] / 6 haneli seri No. / [ENTER] / [CLEAR]: 32. **Zone**a kablosuz ürün tanıtımı

Kablosuz hırsız alarm santraline dedektör, manyetik kontak veya PGM modülünün seri numarası eklendikten sonra eklenen ürünün hırsız alarm santrali ile bağlantı-sinyal gücünün görülmesi gerekir. Buna göre sinyal seviyesi düşük ise ilgili ürünün yeri değiştirilmelidir. Bir anlamda bu şekilde tanıtımın kontrol edilmesi de sağlanır.

Kablosuz ürünün sinyal gücü ölçümü için önce programlama menüsüne mühendislik kodu ile girilir. **Zone** numarası girildikten sonra ürünün üzerindeki **TAMPER** anahtarına basılır. **Keypad** üzerindeki ışıkların yanışına ve uyarı sinyaline göre sinyal seviyesi belirlenir. Örneğin **keypad**de 8-10 arası ışıklar yanıp/**3 beep** sesi alınıyor ise en iyi sinyal, 5-7 arası ışıklar yanıp/**2 beep** sesi alınıyor ise orta sinyal, 1-4 arası ışıklar yanıp/**1 beep** sesi alınıyor ise zayıf sinyal anlamına gelir.

Örnek 3: ENTER/[0000]/[101]: 1. **Zone**un sinyal gücünü ölçmek için kodu girin, **TAMPER** anahtarına basın.

Örnek 4: ENTER/[0000]/[132]: 32. **Zone**un sinyal gücünü ölçmek için kodu girin, **TAMPER** anahtarına basın.

Kablosuz hırsız alarm santraline kablosuz siren tanıtımı diğer kablosuz ürünlerde olduğu gibi yapılır. Aralarındaki farklılık, diğer kablosuz ürünlerde seri numarasının 6 haneli, kablosuz sirenlerde ise 4 haneli olmasıdır. Kablosuz ürünler 32 adede kadar tanıtılabilirken kablosuz sirenler genellikle 4 adede kadar tanıtılabilir. Bir de kablosuz sirenlerin tanıtımı ve sinyal gücünün görülmesi için gereken kodlar farklıdır.

Örnek 1: ENTER / [0000] / [683] / 4 haneli seri No. / [ENTER] / [CLEAR]: 1. Sirenin kablosuz tanıtımı

Örnek 2: ENTER / [0000] / [686] / 4 haneli seri No. / [ENTER] / [CLEAR]: 4. Sirenin kablosuz tanıtımı

Örnek 3: ENTER/[0000]/[687]: 1. Sirenin sinyal gücünü ölçmek için kodu girin TAMPER anahtarına basın.

Örnek 4: ENTER/[0000]/[690]: 4. Sirenin sinyal gücünü ölçmek için kodu girin, TAMPER anahtarına basın.

5.3.9. Uzaktan Kumandanın Programlanması

Hırsız alarm santrallerinde uzaktan kumanda kullanımı ile kurma, çözme vb. işlemlerin yapılabilmesi için uzaktan kumandanın tanıtımı sağlanır. Uzaktan kumanda tanıtımı da kablosuz ürünlerin tanıtımına çok benzerdir, sadece tanıtım kodları farklıdır. Hırsız alarm santraline 32 adet uzaktan kumanda eklenebilir. Eklenen uzaktan kumandayı silmek için kablosuz ürünlerde olduğu gibi 6 haneli kod yerine 000000 girilir.

Örnek 1: ENTER / [0000] / [651] / 6 haneli seri No. / [ENTER] / [CLEAR]: 1. Uzaktan kumanda tanıtımı

Örnek 2: ENTER / [0000] / [682] / 6 haneli seri No. / [ENTER] / [CLEAR]: 32. Uzaktan kumanda tanıtımı

5.3.10. Kullanıcı Şifrelerinin Programlanması

Hırsız alarm sistemlerinin montajı, bağlantıları ve programlanması tamamlandıktan sonra sistem kullanıcıya teslim edilir. Kullanıcıya teslim edildiğinde bazı programlama menüleri kullanıcıya kılavuz hâlinde verilebilir. Bu kılavuzda genellikle kullanıcı kodu ekleme, değiştirme, silme, tarih ve saat ayarı, sistemi kurma ve devre dışı bırakma ile ilgili bilgiler yer alır.

Hırsız alarm santrallerinde kullanılan modele göre genellikle 32 kullanıcıya şifre verilmesi sağlanabilir. Bu şifre ile hırsız alarm santralinin kurma ve açılma işlemi gerçekleştirilir. Kullanıcı şifresi tanımlarken ilgili tuş takımı ile önce **master** kodu, sonra kullanıcı sıra numarası girilir. Kullanıcı şifresi 4 veya 6 haneli olarak iki defa girilip onay ve çıkış işlemi yapılır. Kullanıcı şifresi silinirken ise ilgili tuş takımı ile **master** kodu girilir. Sonra kullanıcı sıra numarası girilip ilgili tuş kombinasyonuna onay sesi alınana kadar basılır. 1. örnekteki gibi kullanıcı kodu verme veya değiştirme işlemi yapılabilir. Kullanıcı şifresinin iptal işlemi de 2. örnekte verildiği gibi yapılır.

Örnek 1: [ ] > [**MASTER** KODU] > [Kullanıcı Sırası] > [YENİ KOD] > [YENİ KOD]>[ENTER]>[CLEAR]

Örnek 2: [ ] > [**MASTER** KODU] > [Kullanıcı Sırası] > [SLEEP] tuşuna onay sesi duyana kadar basınız.

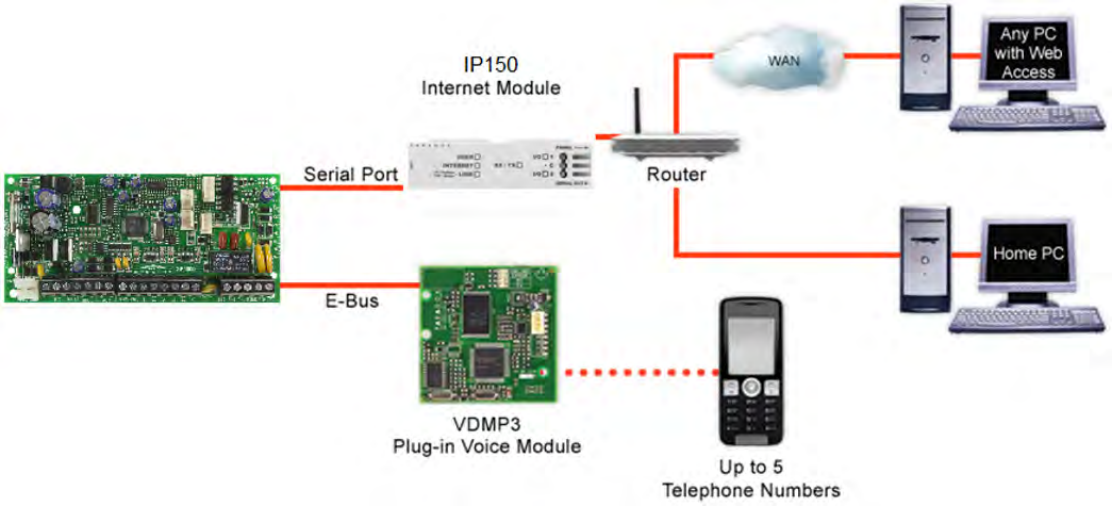
Hırsız alarm santrallerinin kurma ve devre dışı bırakma işlemleri, kullanılan modele göre farklılık gösterir. Genel olarak kullanıcıya normal kurma, evde kurma, tek tuş ile normal kurma, **zone** bypaslama, devre dışı bırakma, panik alarmı ve telefon numarası ekleme ile ilgili bilgiler verilir. Aynı zamanda **keypad** üzerindeki hata mesajlarının nasıl okunacağı yani alarm görüntüleme ile ilgili bilgiler de verilir. Bunların yanında **master** kodu değiştirme yöntemi de gösterilir.

5.3.11. Haber Alma Merkezinin Programlanması

Hırsız alarm santrali üreticileri, telefon veya internet üzerinden takip edebilmek için geliştirdikleri santrallere erişim imkânı sunmuşlardır. Kullanılan hırsız alarm santralinin modeline göre haber alma merkezine bağlanma veya telefon hattı üzerinden bağlantı programları farklılık göstermektedir. Kullanılan ürünün programlama sayfalarına bakılarak işlemler gerçekleştirilebilir.

Temel olarak hırsız alarm santrallerine, **master** kodu ile alarm durumunda aranması için 5 ile 8 adet telefon numarası eklenebilmektedir. Ekstra gözlem istasyonu programlaması ve IP/GPRS haberleşme programlaması mühendislik kodu ile yapılabilmektedir. Haber alma merkezi **reciver** numarası olarak 2 ile 4 telefon numarası mühendislik kodu ile eklenebilir.

Görsel 5.50'de hırsız alarm santral kartının haber alma merkezi ile sinyal akış diyagramı görülmektedir. Bu görsel de network, telefon ve GSM iletişimine örnek olarak verilebilir.



Görsel 5.50: Haber alma merkezinin sinyal akış diyagramı

5.3.12. İOS-Android Yazılımların Programlanması

Hırsız alarm santralleri üreticileri, geliştirdikleri ürünlere özel aplikasyon yazılımları üretmektedir. Bu aplikasyon yazılımlarını İOS-Android uygulamalar için geliştirip ilgili yerlere yüklemektedir. Hırsız alarm santrali satın alan ve montajını yaptıran müşteriler, aplikasyon yazılımlarını cep telefonlarına indirip hırsız alarm santralinin takibinde kullanabilmektedir.

Cep telefonuna indirilen İOS-Android yazılım uygulaması; kullanıcı adı, mail, şifre vb. kullanıcı bilgileri girilerek çalıştırılır. Hırsız alarm santrali üzerindeki seri numarası veya fabrikasyon üretilen QR kodu eklenerek İOS-Android uygulama yazılımıyla entegre edilmesi sağlanır. Bu uygulama yazılımlarında genel olarak **keypad** ile kullanıcının yapabildiği bütün işlemler (sistemi kurma ve açma, PGM kontrolü, kameralı PIR'ların izlenmesi vb.) yapılabilmektedir. Aynı zamanda hırsız alarm santrali kurulu iken uzaktan sistemin takip edilmesinde kullanılır. Alarm, hata, kurma ve açma gibi durumlarda anlık bildirimler gönderir. Görsel 5.51'de örnek bir İOS-Android uygulamasının görselleri görülmektedir.



Görsel 5.51: Örnek bir IOS- Android telefonlar ile sistemin uzaktan kontrol uygulaması

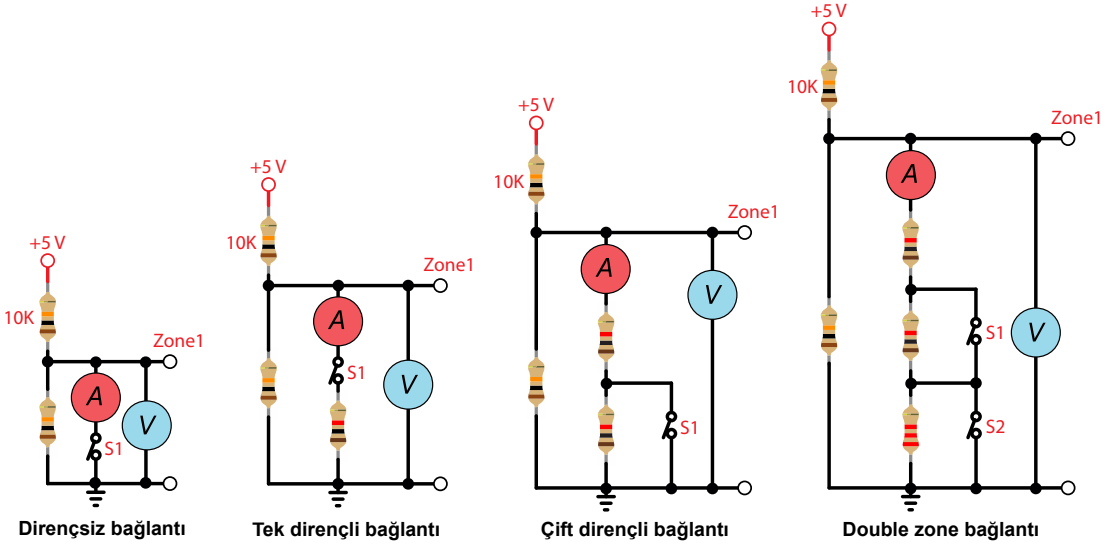
Uygulama Adı: Hırsız Alarm Sistemlerinde Kullanılan Zone Bağlantı Modellemeleri — No.: 1

Amaç: Hırsız alarm sistemlerinde kullanılan zone bağlantı modellemeleri yapmak.

Hırsız alarm santralinin, dirençli ve dirençsiz **zone** bağlantılarının çalışma mantığının anlaşılabilmesi için açık kontak (açık devre), kapalı kontak (kısa devre), dirençler ile yapılmış seri devrede gerilim hesaplama ve ölçme konularının bilinmesi gerekir. Kirşof gerilimler ve akımlar kanununun hatırlanması hırsız alarm sisteminin çalışmasının anlaşılmasını kolaylaştıracaktır.

Görsel 5.53'te hırsız alarm sistemlerinde kullanılan bağlantı türleri ve ölçümleri görülmektedir. Buradaki S1 ve S2 anahtarları hırsız alarm sistemlerinde kullanılan dedektörlerin yerine kullanılmıştır. +5 V kaynağa seri bağlı **10K** dirençler hırsız alarm santralinin kartının üzerindeki **zone** uçlarının bağlantı şeması olarak görülmelidir.

Uygulamanın işlem basamaklarında verilen Tablo 5.2 ve Tablo 5.3'teki direnç (R) ölçümleri ilgili devre şemasında enerji verilmeden yapılmalıdır.



Görsel 5.53: Hırsız alarm sistemlerinde kullanılan bağlantı türleri ve ölçümleri

İşlem Basamakları

1. Öğretmeninizden devre malzemelerinin kontrolünü yaparak alınız.
2. Görsel 5.53'teki dirençsiz, tek dirençli ve çift dirençli devre şemalarının bağlantılarını ayrı ayrı kurarak Tablo 5.2'deki ölçüm sonuçlarını ilgili yerlere yazınız.
3. Görsel 5.53'teki **double zone** devre şemasının bağlantısını kurarak Tablo 5.3'teki ölçüm sonuçlarını ilgili yerlere yazınız.
4. Öğretmeninize devre malzemelerinin kontrolünü yaparak teslim ediniz.

Tablo 5.2: Hırsız Alarm Sistemlerinde Kullanılan Bağlantı Türlerinde Akım-Gerilim-Direnç Ölçümleri

Bağlantı Türleri	S1 ve S2 Anahtarlarının Durumu		Zone1-GND (R)	Zone1-GND (V)	Zone1-GND (A)	Hırsız Alarm Santrali Çalışma Modları
Dirençsiz bağlantı	S1	AÇIK				
		KAPALI				
Tek Dirençli Bağlantı	S1	AÇIK				
		KAPALI				
Çift Dirençli Bağlantı	S1	AÇIK				
		KAPALI				

Tablo 5.3: Hırsız Alarm Sistemlerinde Kullanılan Double Zone Bağlantı Türünün Ölçümleri

Bağlantı Türleri	S1 ve S2 Anahtarlarının Durumu S1	Zone1-GND (R) S2	Zone1-GND (V)	Zone1-GND (A)	Hırsız Alarm Santrali Çalışma Modları
DoubleZone Bağlantı					
	AÇIK	AÇIK			
	AÇIK	KAPALI			
	KAPALI	AÇIK			
	KAPALI	KAPALI			

Sorular

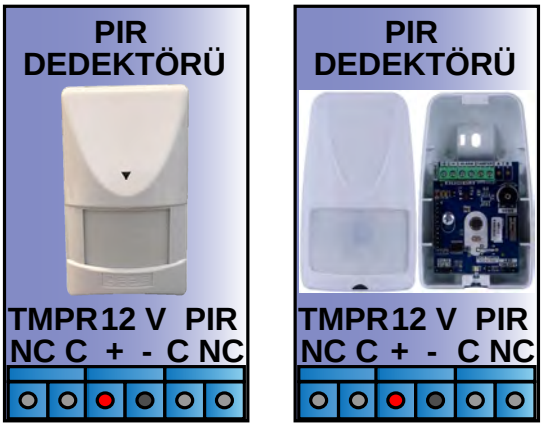
1. Tablo 5.2'deki sonuçlara göre hırsız alarm santralinin çalışma modlarını dikkate alarak bağlantı türlerine göre çalışma mantığını açıklayınız.
2. Tablo 5.3'teki sonuçlara göre hırsız alarm santralinin çalışma modlarını dikkate alarak bağlantı türlerine göre çalışma mantığını açıklayınız.

Uygulama Adı: Hırsız Alarm Sistemlerinde Kullanılan Dedektörler No.: 2

Amaç: Hırsız alarm sistemlerinde kullanılan dedektörleri tanıtmak.

Hırsız alarm sistemlerinde kullanılan dedektörler, algıladığı unsuru belirtmek için çıkışında röle kontağı (NC-C-NO) gibi kuru kontak anahtarlaması yapar. Çıkışında genellikle NC-C kapalı kontağı kullanılır. Dedektör ilgili unsuru algılamadığında NC-C arası kısa devredir. Dedektör ilgili unsuru algıladığında ise NC-C arası açık devre olur.

Hırsız alarm santrallerinde çıkışı kuru kontak anahtarlaması gibi röle anahtarlaması olan bütün dedektörler kullanılabilir. Kablolu dedektörlerin besleme gerilimleri genellikle AC/DC 12 V'la çalışmaktadır. Manyetik kontak mıknatıslanma prensibine göre panik buton, bas-çek mantığında kuru anahtarlama prensibinde çalıştığı için enerji ihtiyacı duymaz. Görsel 5.54'te PIR dedektörünün, Görsel 5.55'te ise hırsız alarm sistemlerinde kullanılan dedektörlerden bazılarının görselleri görülmektedir.



Görsel 5.54: Hırsız alarm sistemlerinde kullanılan PIR dedektörü

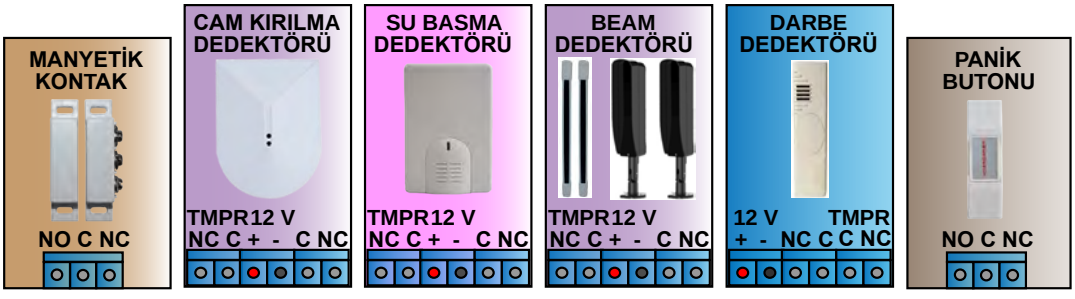
İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak PIR dedektörünün kapağını açınız ve DC 12 V besleme gerilimi bağlantısını ilgili klemenslere yapınız.
2. Öğretmeninizin kontrolünde güç kaynağını çalıştırarak PIR dedektörünün çalışma gerilimini uygulayınız.
3. PIR dedektörünün kapağı açık ve kapalı iken ortamdaki unsuru (hareket) ne kadar mesafeden algıladığına dikkat ediniz.
4. PIR dedektörünün kapağını açarak ortamda hareket algılaması yok iken ve hareket algılaması var iken kuru kontak uçlarını NC-C ölçü aletinin ohm kademesi ile ölçünüz. Tablo 5.4'teki ilgili hücrelere değerleri not ediniz.
5. PIR dedektörünün kapağı açık durumda, ortamda hareket algılaması yok iken ve hareket algılaması var iken **TAMPER** uçlarını NC-C ölçü aletinin ohm kademesi ile ölçünüz. Tablo 5.4'teki ilgili hücrelere değerleri not ediniz.
6. **TAMPER** anahtarının üzerindeki mandala basarak (PIR dedektörünün kapağı kapalı modellemesi) ortamda hareket algılaması yok iken ve hareket algılaması var iken **TAMPER** uçlarını NC-C ölçü aletinin ohm kademesi ile ölçünüz. Tablo 5.4'teki ilgili hücrelere değerleri not ediniz.
7. Güç kaynağının enerjisini kesiniz. PIR dedektörünün besleme gerilimi bağlantılarını sökünüz.
8. PIR dedektörünün kapağını kapatarak öğretmeninize teslim ediniz.

Tablo 5.4: Hırsız Alarm Sistemlerinde Kullanılan PIR Dedektörünün Ölçüm Sonuçları

Dedektörler	Kuru Kontak Uçları NC-C Arası		TAMPER			
			Switchin Mandalı Açık İken		Switchin Mandalı Kapalı/Basılı İken	
	Ortamda Hareket Yok İken	Ortamda Hareket Var İken	Ortamda Hareket Yok İken	Ortamda Hareket Var İken	Ortamda Hareket Yok İken	Ortamda Hareket Var İken
PIR Dedektörü						

1. İşlem basamaklarını Görsel 5.55'teki hırsız alarm sistemlerinde kullanılan bütün dedektörler için ayrı ayrı uygulayarak Tablo 5.5'teki ilgili yerlere ölçüm sonuçlarını yazınız.



Görsel 5.55: Hırsız alarm sistemlerinde kullanılan dedektörlerden bazıları

Tablo 5.5: Hırsız Alarm Sistemlerinde Kullanılan Dedektörlerin Ölçüm Sonuçları

Dedektörler	Kuru Kontak Uçları NC-C Arası		TAMPER			
			Switchin Mandalı Açık İken		Switchin Mandalı Kapalı/Basılı İken	
	Ortamda Hareket Yok İken	Ortamda Hareket Var İken	Ortamda Hareket Yok İken	Ortamda Hareket Var İken	Ortamda Hareket Yok İken	Ortamda Hareket Var İken
Cam Kırılma Dedektörü						
Su Basma Dedektörü						
Beam Dedektörü						
Darbe Dedektörü						
Manyetik Kontak						
Panik Butonu						

Sorular

1. PIR dedektörüne enerji verildiğinde kapağı kapalıyken mi yoksa açıkken mi daha uzaktan hareket algılar? Sebebi ile birlikte açıklayınız.
2. Manyetik kontağın çalışma prensibini açıklayınız.
3. Manyetik kontağın içinde hangi elektronik malzeme kullanılmış olabilir? Araştırınız.

NOT DEFTERİ

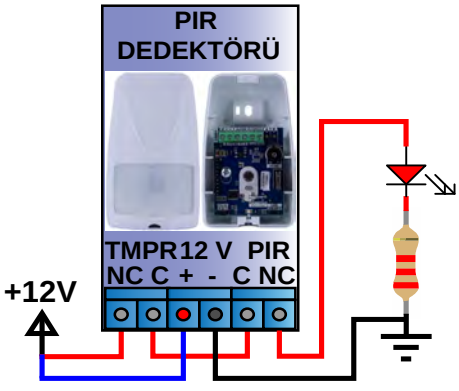
Uygulama Adı: Hırsız Alarm Sistemlerinde Kullanılan Dedektörlerle Lamba Uygulaması — No.: 3

Amaç: Hırsız alarm sistemlerinde kullanılan dedektörlerle lamba uygulaması yapmak.

Hırsız alarm sistemlerinde kullanılan dedektörlerin çıkışında NC-C kontak bağlantı uçları olduğu gibi bazen de NC-C-NO gibi bağlantı uçları bulunur.

Görsel 5.56'da PIR dedektörünün çıkışına bağlanan lamba uygulama şeması görülmektedir. Bu bağlantı NC-C kontakları arasında yapılmış olup aynı zamanda hırsız alarm santralinin çalışma mantığını ön plana çıkarmak için gerçekleştirilmiştir. Fakat sahada PIR dedektörüyle yapılan lamba uygulamalarında NO-C arasına bağlantı yapılmaktadır. İki bağlantı arasındaki fark, birinde PIR dedektörünün hareketi algıladığında lambanın sönmesi, diğerinde ise PIR dedektörünün hareketi algıladığında lambanın yanmasıdır.

Bu uygulama, hırsız alarm sistemlerinde kullanılan bütün dedektörler için yapılabilir.



Görsel 5.56: PIR dedektörünün çıkışına bağlanan lamba uygulaması

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak PIR dedektörünün kapağını açınız ve Görsel 5.56'daki devre şemasının bağlantısını kurunuz.
2. Öğretmeninizin kontrolünde devre şemasına DC 12 V besleme gerilimini güç kaynağından uygulayınız.
3. Sistemin çalışmasını gözlemleyiniz. PIR dedektörünün içindeki PIR sensörünün önünde hare ket ediniz. LED diyot yanıyor mu? LED diyot yanmıyor ise sebebi nedir? Açıklayınız.
4. PIR dedektörünün kapağını kapatınız veya PIR dedektörünün kapağını kapalı duruma getirmek için **TAMPER** anahtarının mandalını basılı duruma getiriniz.
5. 3. işlem basamağını tekrarlayınız.
6. Güç kaynağının enerjisini kesiniz. PIR dedektörünün bağlantılarını sökünüz.
7. Aynı işlem basamaklarını hırsız alarm sistemlerinde kullanılan diğer dedektörler ve manyetik kontak için de uygulayınız.
8. Kullandığınız malzemelerin kontrolünü yaparak öğretmeninize teslim ediniz.

Sorular

1. PIR dedektörünün nasıl çalıştığını açıklayınız.
2. **TAMPER switch**in görevini açıklayınız.
3. PIR dedektörünün kapağının işlevini/özelliğini açıklayınız.
4. PIR dedektörünün hareket algıladığı zaman LED diyot yanacak şekilde Görsel 5.56'daki devre şemasının bağlantısını güncelleyiniz ve uygulamasını yapınız.
5. AC 220 V PIR dedektörü devre şemasını araştırınız.
6. Arduino ile PIR sensörü uygulamasını bilgisayar simülasyon programında tasarlayınız.

NOT DEFTERİ

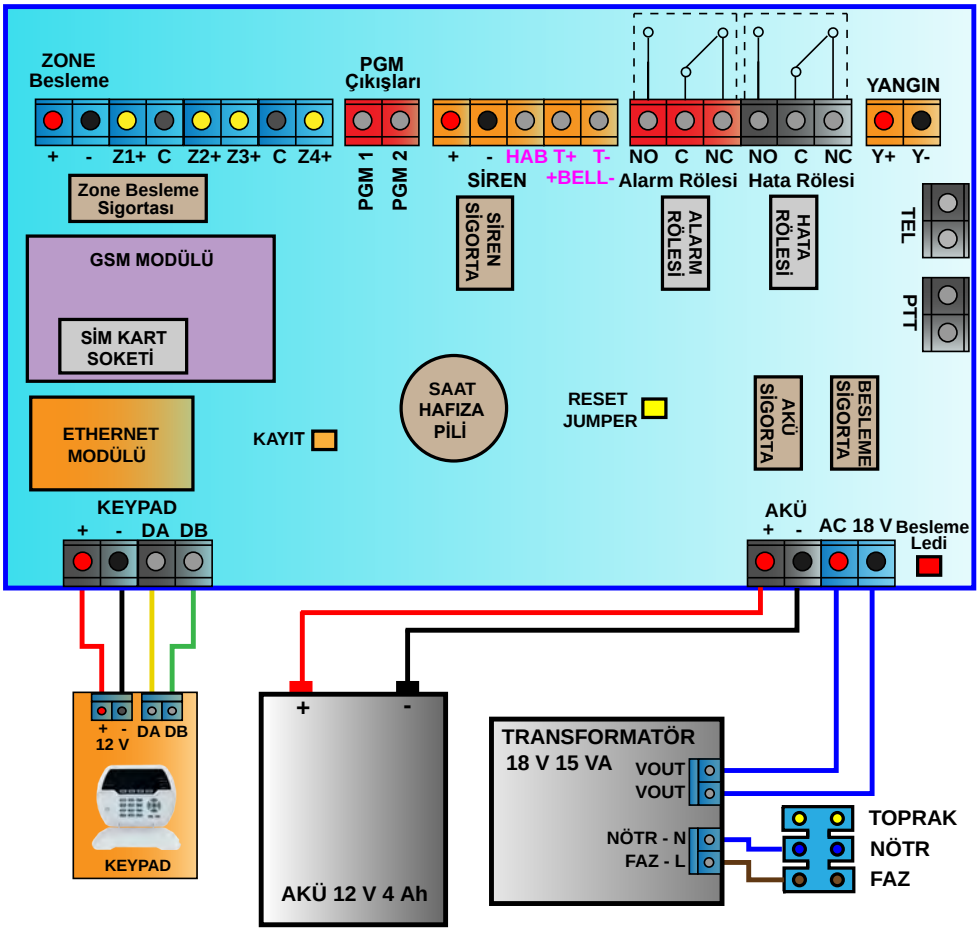
Uygulama Adı: Kablolu Hırsız Alarm Santralinin Resetlemesi ve Tarih ile Saat Ayarı — No.: 4

Amaç: Kablolu hırsız alarm santralinin resetlenmesini ve tarih ile saat ayarını yapmak.

Hırsız alarm santralleri ilk alındığında fabrikasyon olarak resetlenmiştir. Yani hiçbir **zone** tanımlı değildir. Hırsız alarm santralleri ile çalışmaya başlamadan önce mutlaka hırsız alarm santrali resetlenmelidir. Hırsız alarm santrallerinin resetlenmesi tercih edilen modele göre farklılık göstermektedir. Hırsız alarm santrallerinin nasıl resetleneceği ilgili ürün kataloğunda üretici tarafından belirlenmiştir.

Hırsız alarm santrallerinin çalıştırabilmesi ve programlanabilmesi için mutlaka **keypad** bağlantısının yapılması gerekir. Bazı hırsız alarm santrallerinde tarih ve saat ayarı yapılmadan **zone** programlamaları yapılamamaktadır. Tarih ve saat ayarlarının nasıl yapılacağı tercih edilen modele göre farklılık göstermektedir. Hırsız alarm santrallerinde tarih ve saat ayarının nasıl yapılacağı ilgili ürün kataloğunda üretici tarafından belirlenmiştir.

Görsel 5.57'de kablolu hırsız alarm santralinin resetlenebilmesi ve tarih ile saat ayarının yapılabilmesi için bağlantı klemens şeması verilmiştir.



Görsel 5.57: Kablolu hırsız alarm santralinin bağlantı klemens yapısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak hırsız alarm santralinin kapağını açınız. Görsel 5.57'deki bağlantı şemasını kurunuz.
2. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
3. Hırsız alarm santralini **keypad**den girilen program kodları ile resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
4. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarını yapınız.
5. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarının yapıldığını teyit ediniz.
6. Hırsız alarm santrali üzerindeki buton veya **jumper** ile resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
7. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarının yapıldığını teyit ediniz.
8. Hırsız alarm santralini **zone** uçlarına yapılan kablo bağlantısı ile resetleyiniz. Resetleme sonunda sistem devrede LED'inin yandığını gözlemleyiniz.
9. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Sorular

1. Kullandığınız veya kullanacağınız hırsız alarm santrallerinin modellerini yazınız.
2. Kullandığınız veya kullanacağınız hırsız alarm santrallerinin modeline göre resetleme yöntemlerini maddeler hâlinde yazınız.
3. Kullandığınız veya kullanacağınız hırsız alarm santrallerinin modeline göre tarih ve saat ayarlarının nasıl yapıldığını açıklayınız.

NOT DEFTERİ

Uygulama Adı: Kablolu Hırsız Alarm Santralinde Zone Programlama ve Zone Ölçümleri — No.: 5

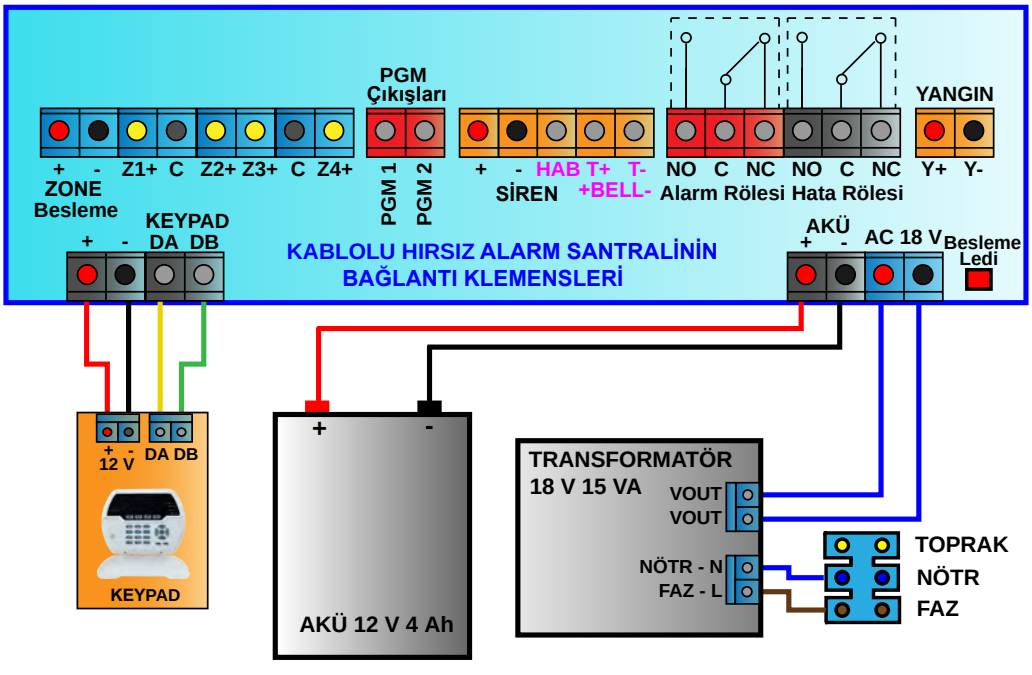
Amaç: Kablolu hırsız alarm santralinde zone programlama ve zone ölçümleri yapmak.

Hırsız alarm santrallerinde **zone** özellikleri program kodları ile değiştirilebilmektedir. Hırsız alarm santrallerinin bazı modellerinde **zone**ların dirençli bağlantı, dirençsiz bağlantı ve **double zone** bağlantı olarak kullanılması program kodları ile ayarlanabilir. Görsel 5.58'de kablolu hırsız alarm sisteminin bağlantı klemensleri ve manuel çalıştırılması görülmektedir.

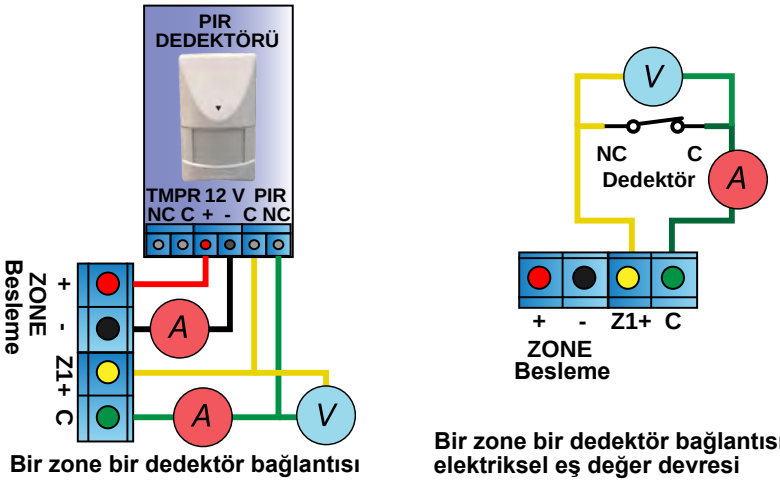
Hırsız alarm santrallerinin programlanması, dirençli, dirençsiz ve **double zone** bağlantı şemaları, kullanılacak dirençlerin değerleri kullanılan hırsız alarm santraline göre farklılık gösterebilir. Temrinlerde verilen uygulama şemalarındaki direnç değerleri, bağlantı şemaları ve programlama kodları için kullanacağınız hırsız alarm santralinin kataloğundaki değerleri tercih ediniz. Hırsız alarm santrallerinin **zone** bağlantılarında kullanılacak dirençlerin ilgili dedektörün üzerinde bağlantısını gerçekleştiriniz. Görsel 5.59'da dirençsiz bağlantılı, Görsel 5.60'ta ise dirençli bağlantılı kablolu hırsız alarm santralinin manuel çalıştırılmasında kullanılan bağlantı şeması ve elektriksel eş değer devresi görülmektedir.

NOT

Uygulama aşamalarında hırsız alarm santralinin **zone** bağlantı türüne göre **zone** özelliklerini ve bağlantı türünü programlamayı unutmayınız. Tablo 5.6'daki hareketsiz ortam ölçümlerini kolaylıkla yapabilmek için PIR dedektörünün önünü sabit bir nesne ile kapatınız.



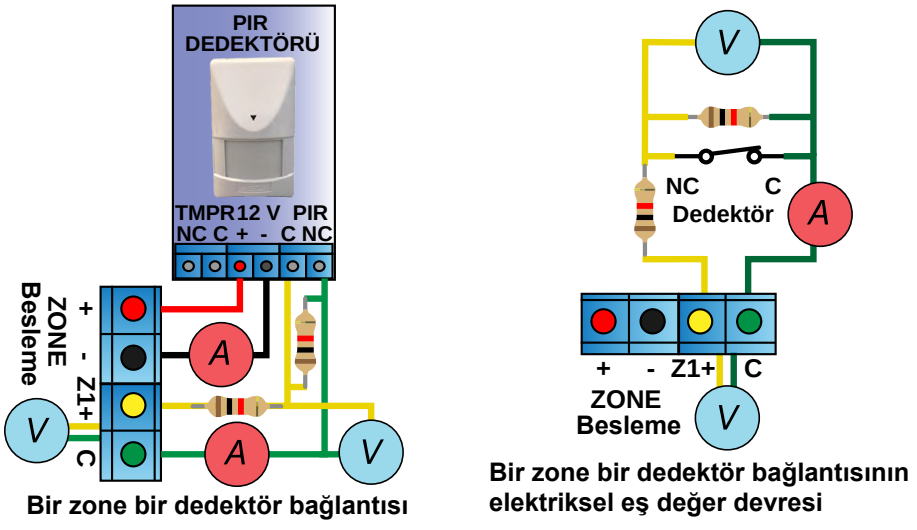
Görsel 5.58: Kablolu hırsız alarm santralinin manuel çalıştırılması



Görsel 5.59: Dirençsiz bağlantılı kablolu hırsız alarm santralinin manuel çalıştırılması

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak hırsız alarm santralinin kapağını açınız. Görsel bağlantı şemasını kurunuz.
2. Görsel 5.59'daki bağlantı şemasını Görsel 5.58 üzerinde gerçekleştiriniz.
3. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
4. Hırsız alarm santralini resetleme yöntemlerinden biri ile resetleyiniz. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarını yapınız.
5. **Keypad** aracılığı ile hırsız alarm santralinin 1 numaralı **zone**unu "normal (giriş gecikmeli) **zone**" olarak ilgili program kodlarını girerek tanıtınız.
6. Tablo 5.6'da verilen ölçümleri yaparak değerleri ilgili hücreye not ediniz.
7. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.



Görsel 5.60: Dirençli bağlantılı kablolu hırsız alarm santralinin manuel çalıştırılması

8. Görsel 5.59'daki bağlantı şemasını Görsel 5.58 üzerinden sökünüz.
9. Görsel 5.60'taki bağlantı şemasını Görsel 5.58 üzerinde gerçekleştiriniz.
10. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
11. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip tarih ve saat ayarını yapınız.
12. **Keypad** aracılığı ile hırsız alarm santralinin 2 numaralı **zone**unu "normal (giriş gecikmeli) **zone**" olarak ilgili program kodlarını girerek tanıtınız.
13. Tablo 5.6'da verilen ölçümleri yaparak değerleri ilgili hücreye not ediniz.
14. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Tablo 5.6: Hırsız Alarm Sistemlerinde Zone Bağlantı Türüne Göre Ölçüm Sonuçları

Bağlantı Türü	Ortamın Durumu	PIR Akımı	PIR Kontak Gerilimi	Zone Akımı	Zone Gerilimi	Hırsız Alarm Santralinin Çalışma Modu
Dirençsiz Bağlantı	Hareket Var					
	Hareket Yok					
Dirençli Bağlantı	Hareket Var					
	Hareket Yok					

Sorular

1. Dirençli bağlantı ile dirençsiz bağlantıyı Tablo 5.6'da yaptığınız ölçümlere göre karşılaştırınız.
2. Hırsız alarm santrallerinden dirençli bağlantılı olanı mı, dirençsiz bağlantılı olanı mı kullanmayı tercih edersiniz? Sebepleriyle birlikte açıklayınız.
3. Uygulama sırasında kullandığınız hırsız alarm santrallerinin modelini, **zone** bağlantı türünü, direnç değerlerini ve program kodlarını maddeler hâlinde yazınız.

Uygulama Adı: Kablolu Hırsız Alarm Santralinde Siren Bağlantısı ve Ölçümleri — No.: 6

Amaç: Kablolu hırsız alarm santralinde siren bağlantısı ve ölçümleri yapmak.

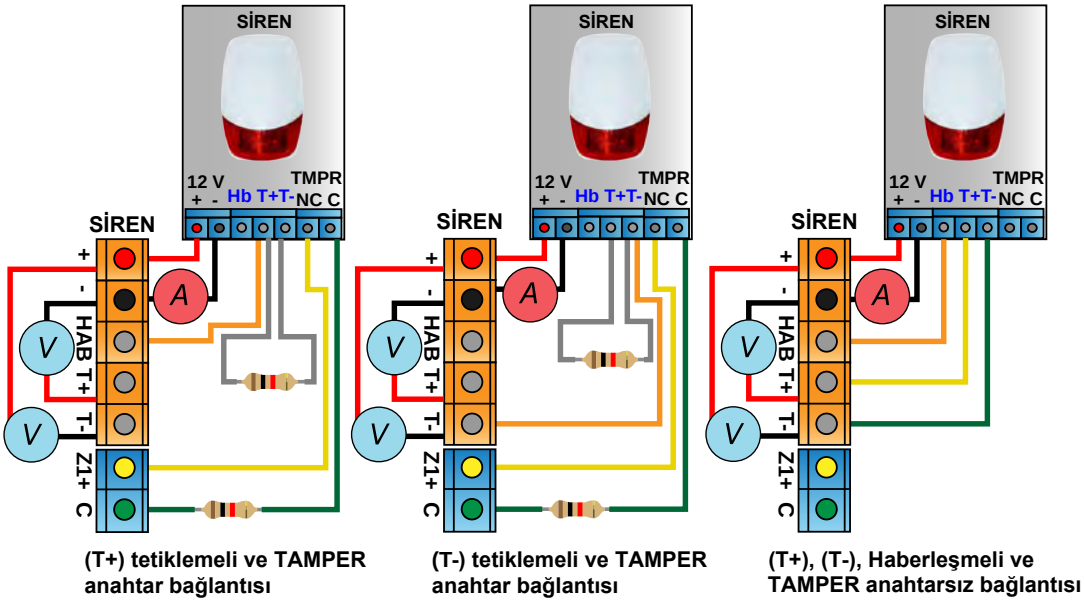
Hırsız alarm santrallerinde **zone**lara dedektör bağlantısı yapılırken dedektör üzerinde bulunan ve kapak **switch**i olarak da bilinen **TAMPER switch** bağlantısı mutlaka yapılmalıdır. **TAMPER switch** bağlantısı ilgili dedektöre yapılacak olan fiziki müdahaleyi algılamak için kullanılır.

Hırsız alarm santralleri; sesli, ışıklı ve sesli-ışıklı sirenleri alarm durumunda insanları uyarabilmek için kontrol eder. Hırsız alarm sistemlerinde kullanılan sirenler +T, -T veya hem +T hem de –T ile hırsız alarm santralleri tarafından alarm durumunda tetiklenebilir. Görsel 5.61'de kablolu hırsız alarm santraline siren bağlantıları görülmektedir.

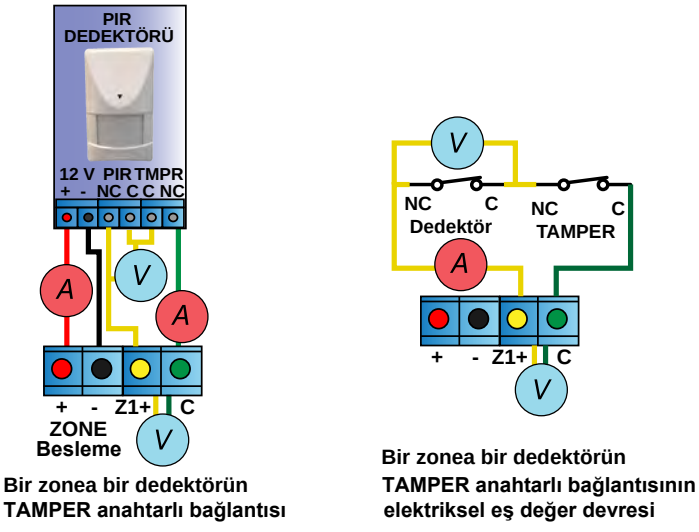
Hırsız alarm sistemlerinde kullanılan sirenlerin bağlantıları, tetikleme (iletime geçirme) yöntemleri tercih edilen modele göre farklılık göstermektedir. Bu özellikler ilgili ürün kataloğunda üretici tarafından belirlenmiştir. Temrinlerdeki siren bağlantı şemasını, tercih ettiğiniz sirenin ürün katalog bilgilerine göre yapınız. Bir **zone**, bir dedektör bağlantısının; Görsel 5.62'de dirençsiz **TAMPER switch** bağlantısı ve elektriksel eş değeri, Görsel 5.63'te ise dirençli **TAMPER switch** bağlantısı ve elektriksel eş değeri görülmektedir.

NOT

Uygulama aşamalarında hırsız alarm santralinin **zone** bağlantı türüne göre **zone** özelliklerini ve bağlantı türünü programlamayı unutmayınız.



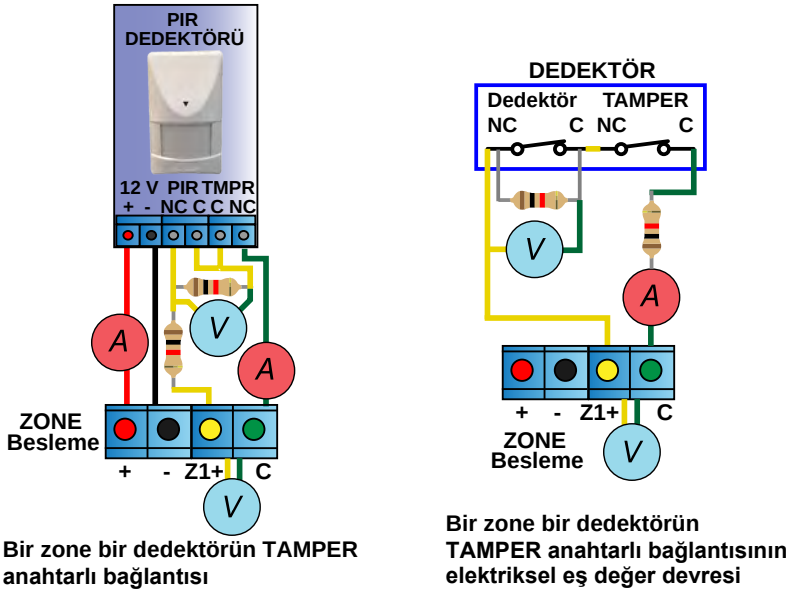
Görsel 5.61: Kablolu hırsız alarm santraline siren bağlantıları



Görsel 5.62: Zonea dirençsiz **TAMPER** switch bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak hırsız alarm santralinin kapağını açınız. Görsel 5.61'deki bağlantı şemalarından ürününüze uygun olanını hırsız alarm santrali üzerinde kurunuz.
2. Görsel 5.62'deki bağlantı şemasını hırsız alarm santrali üzerinde gerçekleştiriniz.
3. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
4. Hırsız alarm santralini resetleme yöntemlerinden biri ile resetleyiniz. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarını yapınız.
5. **Keypad** aracılığı ile hırsız alarm santralinin 1 numaralı **zone**unu "normal (giriş gecikmeli) **zone**" olarak ilgili program kodlarını girerek tanıtınız.
6. Tablo 5.7 ve Tablo 5.8'de verilen ölçümleri yaparak değerleri ilgili hücreye not ediniz.
7. Öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.



Görsel 5.63: Zonea dirençli **TAMPER switch** bağlantısı

8. Görsel 5.62'deki bağlantı şemasını hırsız alarm santrali üzerinden sökünüz.

9. Görsel 5.63'teki bağlantı şemasını hırsız alarm santrali üzerinde gerçekleştiriniz.

10. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.

11. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip tarih ve saat ayarını yapınız.

12. Keypad aracılığı ile hırsız alarm santralinin 2 numaralı **zone**unu "normal (giriş gecikmeli) zone" olarak ilgili program kodlarını girerek tanıtınız.

13. Tablo 5.7 ve Tablo 5.8'de verilen ölçümleri yaparak değerleri ilgili hücreye not ediniz.

14. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Tablo 5.7: Hırsız Alarm Sistemlerinde Zone Bağlantı Türüne Göre Ölçüm Sonuçları

Bağlantı Türü	Ortamın Durumu	PIR Akımı	PIR Kontak Gerilimi	Zone Akımı	Zone Gerilimi	Hırsız Alarm Santralinin Çalışma Modu
Dirençsiz Bağlantı	Hareket var.					
	Hareket yok.					
Dirençli Bağlantı	Hareket var.					
	Hareket yok.					

Tablo 5.8: Hırsız Alarm Sistemlerinde Siren Ölçüm Sonuçları

Bağlantı Türü	Ortamın Durumu	Sirenin Akımı	+T Tetikleme Gerilimi	-T Tetikleme Gerilimi	Hırsız Alarm Santralinin Çalışma Modu
Dirençsiz Bağlantı	Hareket var.				
	Hareket yok.				
Dirençli Bağlantı	Hareket var.				
	Hareket yok.				

Sorular

1. Uygulama sırasında kullandığınız hırsız alarm santrallerinin modelini, **zone** bağlantı türünü, direnç değerlerini ve program kodlarını maddeler hâlinde yazınız.

2. Uygulama sırasında kullandığınız sirenin modelini ve tetikleme yöntemini yazınız.

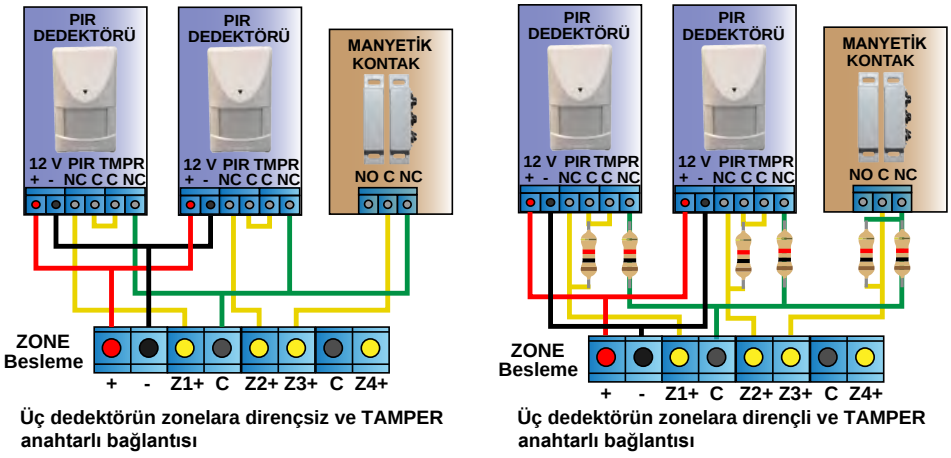
3. Hırsız alarm santrallerinde kullanılan dedektörlerin **zone** bağlantısı yapılırken **TAMPER switch** kullanılmasının sebebini yazınız.

4. Tablo 5.7'deki verileri uygulama 5'teki Tablo 5.6 ile karşılaştırınız.

Uygulama Adı: Üç Dedektör ve Bir Siren Bağlantılı Kablolu Hırsız Alarm Sistemi Kurulumu — No.: 7

Amaç: Üç dedektör ve bir siren bağlantılı kablolu hırsız alarm sistemi kurulumu yapmak.

Görsel 5.64'te dedektörlerin hırsız alarm santrallerine dirençli ve dirençsiz bağlantıları görülmektedir. Temrini uygulama aşamalarında hırsız alarm santralinin **zone** bağlantı türüne göre **zone** özelliklerini ve bağlantı türünü programlamayı unutmayınız.



Görsel 5.64: Üç dedektörün zonelara dirençsiz ve dirençli bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak hırsız alarm santralinin kapağını açınız. Görsel 5.64'daki dirençsiz **zone** bağlantı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.
2. Uygulama 6'da verilen Görsel 5.61'deki siren bağlantı şemalarına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.
3. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
4. Hırsız alarm santralini resetleme yöntemlerinden biri ile resetleyiniz. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarını yapınız.
5. **Keypad** aracılığı ile hırsız alarm santralinin 1 numaralı **zone**unu "ani **zone**" olarak ilgili program kodlarını girerek tanıtınız.
6. **Keypad** aracılığı ile hırsız alarm santralinin 2 numaralı **zone**unu "normal (giriş gecikmeli) **zone**" olarak ilgili program kodlarını girerek tanıtınız.
7. **Keypad** aracılığı ile hırsız alarm santralinin 3 numaralı **zone**unu "24 saat **zone**" olarak ilgili program kodlarını girerek tanıtınız.
8. Giriş gecikmesini 25 sn., çıkış gecikmesini 15 sn., siren çıkış süresini 5 dk. olacak şekilde ilgili program kodlarını girerek tanıtınız.
9. Hırsız alarm santralini hafta içi saat 18.30'da otomatik kurulacak şekilde programlayınız.
10. Hırsız alarm santraline sizin belirleyeceğiniz iki adet kullanıcı şifresi ekleyiniz.
11. Hırsız alarm sistemini ilgili kodları girerek aktif hâle getiriniz ve yaptığınız program kodlarının çalıştığını gözlemleyiniz. Hırsız alarm santralini pasif hâle getiriniz.
12. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

13. Görsel 5.64'teki dirençli **zone** bağlantı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz. İşlem basamaklarını tekrarlayınız.

Sorular

1. Uygulama sırasında kullandığınız hırsız alarm santrallerinin modelini, **zone** bağlantı türünü, direnç değerlerini ve program kodlarını maddeler hâlinde yazınız.

2. Uygulama sırasında kullandığınız sirenin modelini ve tetikleme yöntemini yazınız.

3. Kullandığınız hırsız alarm santralini hafta sonu saat 10.00'da otomatik kurulacak şekilde nasıl programlarsınız? Program kod parçacığını araştırınız ve bulduğunuz sonucu yazınız.

NOT DEFTERİ

Uygulama Adı: Bir Zonea Birden Fazla Dedektörü Seri Bağlama No.: 8

Amaç: Bir zonea birden fazla dedektörü seri bağlama uygulaması yapmak.

Hırsız alarm santrallerinde, alarm durumunda ilgili bölgeye hızlı erişim sağlamak için her **zone**a bir dedektör bağlanması istenir. Aşağıda belirtilen sebeplerden dolayı bir **zone**a birden fazla dedektör bağlanabilir:

• Kullanılacak dedektör sayısının seçilen hırsız alarm santralinin **zone** sayısından fazla olması durumunda bir **zone**a birden fazla dedektör seri olarak bağlanabilir.

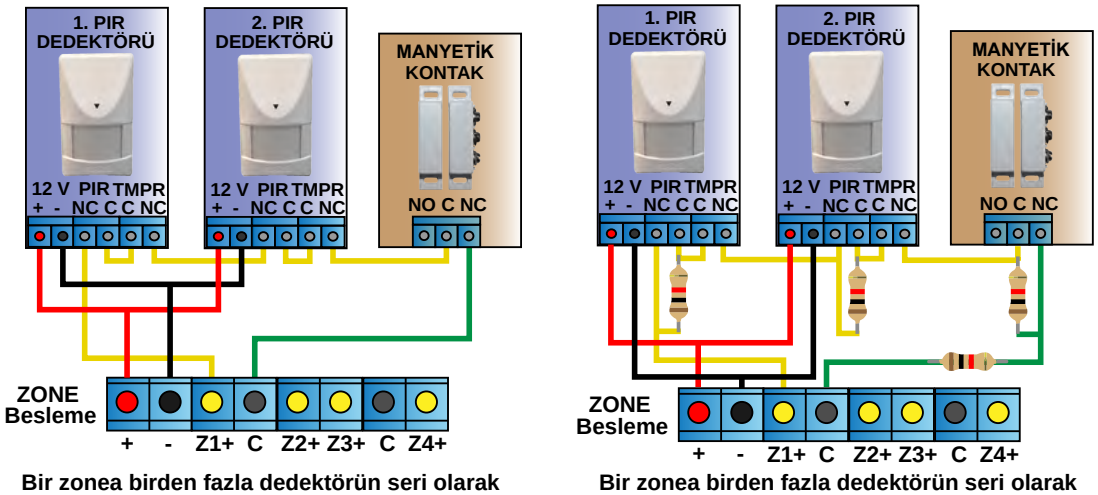
• Hırsız alarm sisteminin montajı sırasında, aynı bölgede bulunan ve birbirine yakın olan dedektörlerin sistemin genel çalışmasını etkilememek kaydıyla kullanılacak kablo ve montaj işçiliğini azaltmak için bir **zone**a birden fazla dedektör seri olarak bağlanabilir.

Bir **zone**a birden fazla dedektör bağlantısı yapılırken aşağıdaki özeliklere dikkat edilmelidir:

• Kullanılacak dedektörlerin hırsız alarm santrallerinin programlanması sırasında aynı **zone** özelliğine sahip olması gerekir. Örneğin bir **zone**a hem PIR dedektörü hem de manyetik kontak seri olarak bağlanıyor ise her iki dedektör de aynı **zone** özelliğinde çalışır. PIR dedektörü giriş gecikmeli **zone** özelliğinde, manyetik kontak 24 saat **zone** özelliğinde olamaz. Her iki dedektör de ya giriş gecikmeli **zone** ya da 24 saat **zone** özelliğinde olabilir. Hırsız alarm sistemlerinin montajı sırasında bu detaya dikkat edilmelidir.

• Kullanılacak dedektörlerin hangisinden "Hırsız var." bilgisi geldiği net değildir. Örneğin bir **zone**a hem PIR dedektörü hem de cam kırılma dedektörü seri olarak bağlanmış olsun. Hırsızlık olayı sırasında iki dedektör de aynı **zone**a bağlandığı için alarm sırasında **keypad**den ilgili **zone**a ait alarm bilgisi görülür fakat alarmın PIR dedektöründen mi yoksa cam kırılma dedektöründen mi geldiği belli olmaz. Alarm veren **zone**a ait bölgeye gidilip iki dedektör de ayrı ayrı yetkililerce incelendikten sonra alarm bilgisini veren dedektör tespit edilebilir.

Görsel 5.65'te hırsız alarm santrallerinde, bir **zone**a birden fazla dedektörlerin dirençli ve dirençsiz seri bağlantıları görülmektedir. Uygulama aşamalarında hırsız alarm santralinin **zone** bağlantı türüne göre **zone** özelliklerini ve bağlantı türünü programlamayı unutmayınız.



Bir zonea birden fazla dedektörün seri olarak dirençsiz ve TAMPER anahtarlı bağlantısı

Bir zonea birden fazla dedektörün seri olarak dirençli ve TAMPER anahtarlı bağlantısı

Görsel 5.65: Bir **zone**a birden fazla dedektörün dirençsiz ve dirençli bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak hırsız alarm santralinin kapağını açınız. Görsel 5.65'teki dirençsiz **zone** bağlantı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.
2. Uygulama 6'da verilen Görsel 5.61'deki siren bağlantı şemalarına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.
3. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
4. Hırsız alarm santralini resetleme yöntemlerinden biri ile resetleyiniz. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarını yapınız.
5. **Keypad** aracılığı ile hırsız alarm santralinin 1 numaralı **zone**unu "ani **zone**" olarak ilgili program kodlarını girerek tanıtınız.
6. Giriş gecikmesini 20 sn., çıkış gecikmesini 10 sn., siren çıkış süresini 3 dk. olacak şekilde ilgili program kodlarını girerek tanıtınız.
7. Hırsız alarm santralini hafta içi saat 18.00'de otomatik kurulacak şekilde programlayınız.
8. Hırsız alarm santraline sizin belirleyeceğiniz dört adet kullanıcı şifresi ekleyiniz.
9. Hırsız alarm sistemini ilgili kodları girerek aktif hâle getiriniz ve yaptığınız program kodlarının çalıştığını gözlemleyiniz. Hırsız alarm santralini pasif hâle getiriniz.
10. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.
11. Görsel 5.65'teki dirençli **zone** bağlantı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz. İşlem basamaklarını tekrarlarken **zone** özelliğini "giriş gecikmeli **zone**" olarak tanımlayınız.

Sorular

1. Uygulama sırasında kullandığınız hırsız alarm santrallerinin modelini, **zone** bağlantı türünü, direnç değerlerini ve program kodlarını maddeler hâlinde yazınız.
2. Uygulama sırasında kullandığınız sirenin modelini ve tetikleme yöntemini yazınız.
3. Hırsız alarm santraline uygulama sırasında eklediğiniz kullanıcı şifrelerinden üçüncü sırada kini yeni belirleyeceğiniz bir şifre ile değiştiriniz. Program kod parçacığını araştırınız ve bulduğunuz sonucu yazınız.
4. Kullandığınız hırsız alarm santralini hafta sonu saat 11.15'te otomatik kurulacak şekilde nasıl programlarsınız? Program kod parçacığını araştırınız ve bulduğunuz sonucu yazınız.
5. Kullandığınız hırsız alarm santraline telefon arayıcı modülü nasıl bağlanır? Araştırınız.

Uygulama Adı: Double Zone Bağlantısı — No.: 9

Amaç: Double zone bağlantısı yapmak.

Hırsız alarm santrallerinde **double zone** yerine, akıllı **zone** veya **zone** çiftlemesi isimleri de kullanılabilir. **Double zone** özelliğinin kullanılabilmesi için tercih edilen hırsız alarm santralinin **double zone** özelliğini desteklemesi gerekmektedir. **Double zone** özelliğinin, **zone** programlaması sırasında aktif hâle getirilmesi gerekir.

Double zone özelliği ile bir **zone** iki farklı **zone** olarak kullanılabilir. Bir **zone**a iki dedektör bağlanır. Hırsız alarm santralinin, alarm durumunda kullanılan dedektörler aynı **zone**a farklı direnç değerlerinde bağlandığı için **keypad**den de farklı numaralarda alarm durumu gözlenir.

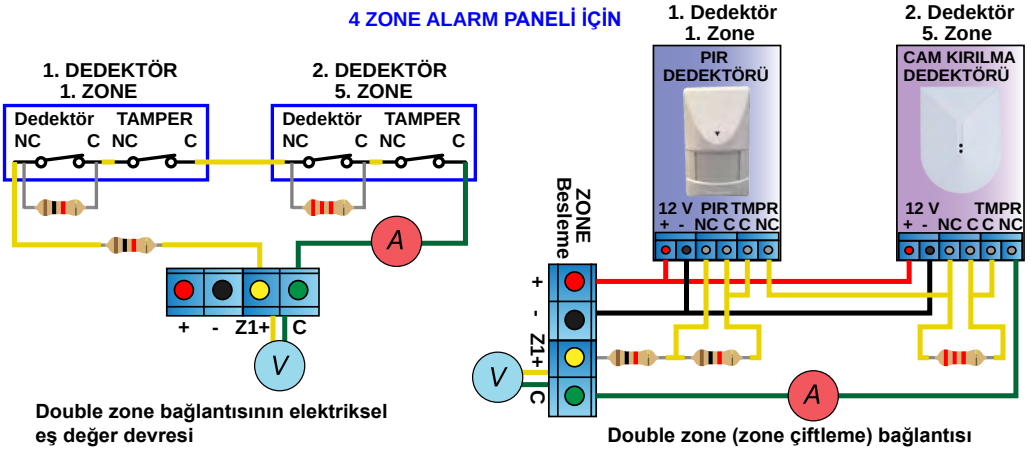
Double zone bağlantı ile seri bağlı bir zonea birden fazla dedektör bağlanması arasındaki farklar şunlardır:

a) Double zoneda her **zone**a iki adet dedektör bağlandığı için maksimum **zone** sayısının iki katı kadar dedektör kullanılabilir. Dedektörlerin seri bağlanmasında böyle bir sayı sınırı yoktur. İstenildiği kadar dedektör seri bağlanabilir.

b) Double zoneda her dedektörün konum bilgisi ve alarm bilgisi bellidir fakat diğerinde belli değildir. Sadece bölge olarak bilinebilir.

c) Her iki bağlantı türünde de ilgili **zone**a bağlanan dedektörlerin özellikleri aynıdır. Yani **zone**lara bağlanan dedektörlerin çalışma özellikleri aynıdır. Farklı programlanamaz.

Görsel 5.66'da hırsız alarm santrallerinde, bir **zone**nun **double zone** bağlantısı ve elektriksel eşdeğeri görülmektedir. Temrini uygulama aşamalarında hırsız alarm santralinin **zone** bağlantı türüne göre **zone** özelliklerini ve bağlantı türünü programlamayı unutmayınız.



Görsel 5.66: Bir **zone**nun **double zone** bağlantısı ve elektriksel eş değeri

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak hırsız alarm santralinin kapağını açınız. Görsel 5.66'daki **double zone** bağlantı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.
2. Uygulama 6'da verilen Görsel 5.61'deki siren bağlantı şemalarına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.

3. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
4. Hırsız alarm santralini resetleme yöntemlerinden biri ile resetleyiniz. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarını yapınız.
5. **Keypad** aracılığı ile hırsız alarm santralinin 1 numaralı **zone**unu "giriş gecikmeli **zone**" olarak ve **double zone** özelliğini aktif edecek şekilde ilgili program kodlarını girerek tanıtınız.
6. Giriş gecikmesini 15 sn., çıkış gecikmesini 5 sn., siren çıkış süresini 2 dk. olacak şekilde ilgili program kodlarını girerek tanıtınız.
7. Hırsız alarm santralini hafta içi saat 17.00'de otomatik kurulacak şekilde programlayınız.
8. Hırsız alarm santraline sizin belirleyeceğiniz beş adet kullanıcı şifresi ekleyiniz.
9. Hırsız alarm sistemini ilgili kodları girerek aktif hâle getiriniz ve yaptığınız program kodlarının çalıştığını gözlemleyiniz. Hırsız alarm santralini pasif hâle getiriniz.
10. Tablo 5.9'daki boş hücreleri hırsız alarm santralini aktif ve pasif hâle getirip ilgili ölçümleri yaparak doldurunuz.
11. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Tablo 5.9: Hırsız Alarm Sistemlerinde Siren Ölçüm Sonuçları

Dedektörlerin Durumu		Zone Eş Değer Direnç Hesabı	Zone Akımı	Zone Gerilimi	Keypadde Yanan Işık Numarası	Hırsız Alarm Santralinin Çalışma Modu
2. Dedektör	1. Dedektör					
Pasif	Pasif					
Pasif	Aktif					
Aktif	Pasif					
Aktif	Aktif					

Sorular

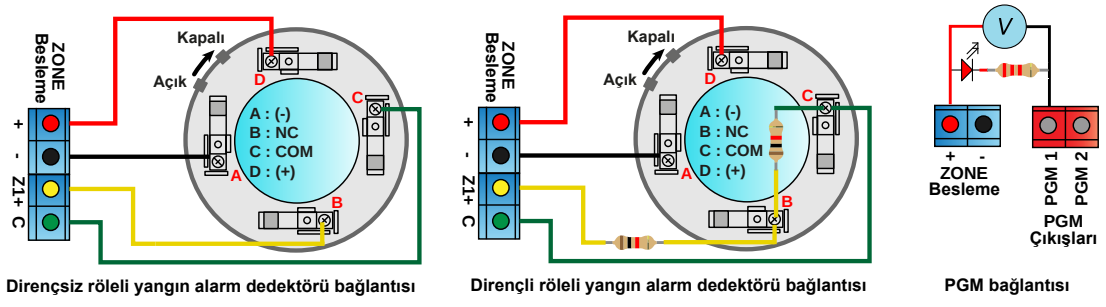
1. Uygulama sırasında kullandığınız hırsız alarm santrallerinin modelini, **zone** bağlantı türünü, direnç değerlerini ve program kodlarını maddeler hâlinde yazınız.
2. Uygulama sırasında kullandığınız sirenin modelini ve tetikleme yöntemini yazınız.
3. Hırsız alarm santraline uygulama sırasında eklediğiniz kullanıcı şifrelerinden dördüncü sıradakini yeni belirleyeceğiniz bir şifre ile değiştiriniz. Program kod parçacığını araştırınız ve bulduğunuz sonucu yazınız.
4. Kullandığınız hırsız alarm santralini hafta sonu saat 10.20'de otomatik kurulacak şekilde nasıl programlarsınız? Program kod parçacığını araştırınız ve bulduğunuz sonucu yazınız.
5. 4 **zone**lu bir hırsız alarm santralinde, 2 numaralı **zone**u **double zone** olarak iki dedektöre bağladığınızda alarm durumunda **keypad**den kaç numaralı ışıklar yanar? Belirtiniz.

Uygulama Adı: Yangın Alarm Dedektörlerinin ve PGM Çıkışlarının Kullanımı — No.: 10

Amaç: Yangın alarm dedektörlerinin ve PGM çıkışlarının kullanım uygulamasını yapmak.

Röle çıkışlı yangın alarm dedektörleri, hırsız alarm sistemlerinde aynı dedektör veya manyetik kontak gibi bağlantısı yapılarak kullanılabilir. Yangın alarm dedektörlerinin hırsız alarm santrallerinde bağlandığı **zone**un özelliğinin "24 saat **zone**" olarak tanımlanması uygun olur.

Hırsız alarm santrallerinde PGM çıkışlarının sayısı seçilen modele göre değişiklik gösterebilir. Hırsız alarm santrallerindeki PGM çıkışları, programlanabilir çıkışlardır. İstenilen **zone**a PGM özelliği dâhil ya da aktif edilebilir. PGM süresi istenilen sürede ayarlanabilir. PGM çıkışları, ilgili **zone** ile programsal olarak tanıtımı yapıldıktan sonra alarm durumunda negatif tetikleme sinyali gönderir. PGM çıkışları maksimum 500 mA akım verebilir. Daha yüksek güçteki ürünleri PGM çıkışı ile kontrol edebilmek için PGM çıkışı ile yüksek güçteki ürünün arasında röle kullanılmalıdır. Görsel 5.67'de hırsız alarm santrallerine, röleli yangın alarm dedektörlerinin dirençli ve dirençsiz bağlantıları ve PGM bağlantısı görülmektedir.



Görsel 5.67: Röleli yangın alarm dedektörü ve PGM bağlantısı

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak hırsız alarm santralinin kapağını açınız. Görsel 5.67'deki PGM bağlantısı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.
2. Görsel 5.67'deki dirençsiz röleli yangın alarm dedektörü **zone** bağlantı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.
3. Uygulama 6'da verilen Görsel 5.61'deki siren bağlantı şemalarına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.
4. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
5. Hırsız alarm santralini resetleme yöntemlerinden biri ile resetleyiniz. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarını yapınız.
6. **Keypad** aracılığı ile hırsız alarm santralinin 1 numaralı **zone**unu "24 saat **zone**" olarak ilgili program kodlarını girerek tanıtınız.
7. Hırsız alarm santralinde 1 numaralı **zone**u alarm durumunda PGM çıkışı olarak aktif olacak şekilde ilgili program kodlarını girerek tanıtınız.
8. Kullandığınız yangın alarm dedektörünün alarm vermesini sağlayınız.
9. PGM uçlarına bağladığınız LED'in yandığını gözlemleyiniz. PGM uçlarının kaç volt çıkış gerilimi verdiğini ölçünüz ve not ediniz (**Ölçülen PGM gerilimi:**.....).
10. Kullandığınız yangın alarm dedektörünün alarm vermesini engelleyiniz.
11. PGM uçlarına bağlanan LED'in söndüğünü gözlemleyiniz. PGM uçlarının kaç volt çıkış gerilimi

verdiğini ölçünüz ve not ediniz (**Ölçülen PGM gerilimi:**.....).

12. Hırsız alarm sistemini ilgili kodları girerek aktif hâle getiriniz ve yaptığınız program kodlarının çalıştığını gözlemleyiniz. Hırsız alarm santralini pasif hâle getiriniz.
13. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.
14. Görsel 5.64'teki dirençli röleli yangın alarm dedektörü **zone** bağlantı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz. İşlem basamaklarını tekrarlayınız.

Sorular

1. PGM çıkışlarına DC 12 V 1 A'lik fan motorunu çalıştıracak devre şemasının tasarımını yapınız.
2. PGM çıkışlarına AC 220 V'luk lambayı çalıştıracak devre şemasının tasarımını yapınız.

NOT DEFTERİ

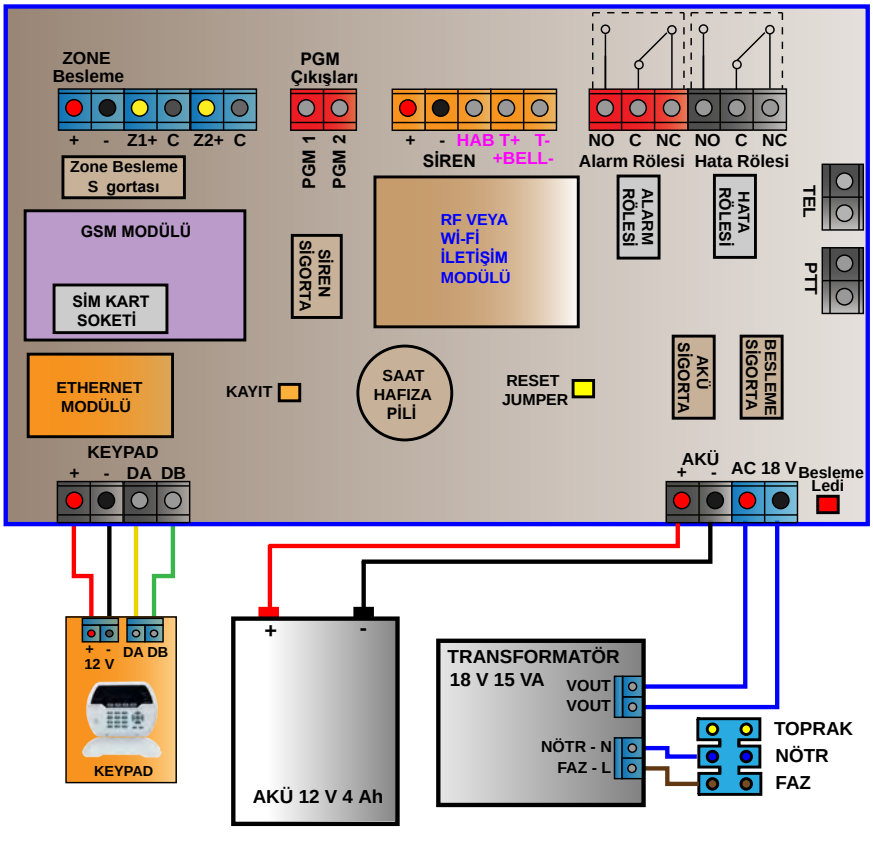
Uygulama Adı: Kablosuz Hırsız Alarm Santralinin Tanıtılması ve Programlanması — No.: 11

Amaç: Kablosuz hırsız alarm santralinin tanıtımını ve programlamasını yapmak.

Hırsız alarm santrallerinin çalıştırabilmesi ve programlanabilmesi için **keypad** bağlantısının mutlaka kablolu yapılması gerekir. Görsel 5.68'de kablosuz hırsız alarm santralinin genel yapısı ve bağlantı klemensleri görülmektedir. Dikkat edilecek olursa kablosuz hırsız alarm santrallerinde de kablolu hırsız alarm santrallerinde olan klemens bağlantı uçları aynen bulunur. Hatta her ihtimale karşılık iki tane de kablolu bağlanabilen **zone** uçları bulunur. Kablosuz hırsız alarm santrallerinde ekstra "**Wi-Fi**" veya "**RF** iletişim modülü" bulunur. İstenir ise hırsız alarm sistemlerinde kullanılan bütün kablosuz ürünler, hırsız alarm santraline **"RF"** veya "**Wi-Fi"** yöntemi ile bağlanabilir. Kablosuz ürünler, kablosuz hırsız alarm santraline tanıtılırken iki yöntem kullanılır.

a) Seri Kod Numarasıyla Tanıtma: Hırsız alarm sisteminde kullanılan ürünlerin içerisinde ürüne ait bir seri kod (ID) vardır. Bu kod panele **keypad** aracılığı ile kullanılan modelin programlama mantığına göre girilerek tanıtılır.

b) Anahtarla Tanıtma: Hırsız alarm sisteminde kullanılan ürünlerin içerisinde bir **dip switch** anahtar/mikrobuton bulunur. Bu mikrobutona basılı tutularak ilgili ürünün hırsız alarm santraline tanıtımı yapılır.

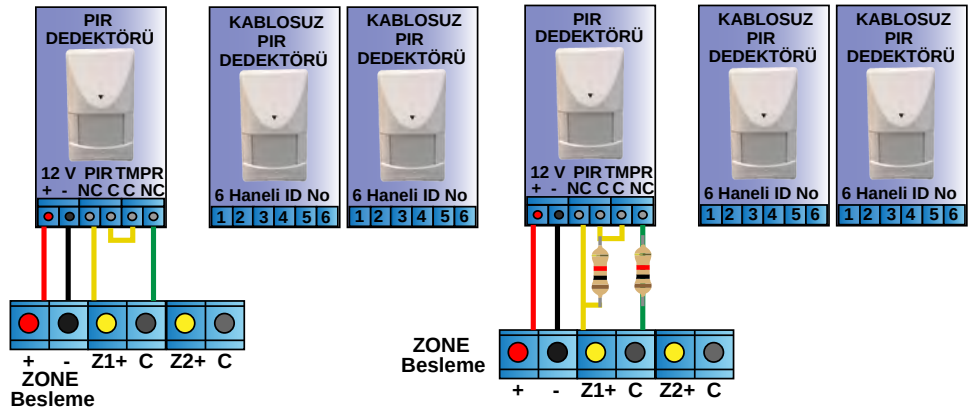


Görsel 5.68: Kablosuz hırsız alarm santralinin genel yapısı ve bağlantı klemensleri

Kablolu hırsız alarm santralleri için yapılan ilk 10 uygulamanın hepsi kablosuz hırsız alarm santrallerinde de kullanılabilir.

İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak hırsız alarm santralinin kapağını açınız. Görsel 5.69'daki bir kablolu dedektörün dirençsiz ve iki kablosuz dedektörün **zone**larına bağlantı şemasını, Görsel 5.68 üzerinde dirençsiz **zone** bağlantı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını belirleyip hırsız alarm santrali üzerinde kurunuz.
2. Uygulama 6'da verilen Görsel 5.61'deki siren bağlantı şemalarına dikkat ederek ürününüze uygun olan bağlantı şemasını hırsız alarm santrali üzerinde kurunuz.
3. Öğretmeninizin kontrolünde şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
4. Hırsız alarm santralini resetleme yöntemlerinden biri ile resetleyiniz. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarını yapınız.



Bir kablolu dedektörün dirençsiz ve iki kablosuz dedektörün zonelara bağlantısı

Bir kablolu dedektörün dirençli ve iki kablosuz dedektörün zonelara bağlantısı

Görsel 5.69: Kablolu-kablosuz dedektörlerin **zone**lara bağlantısı

5. **Keypad** aracılığı ile hırsız alarm santralinin 1 numaralı **zone**una kablo ile bağlantılı PIR dedektörünü "normal (giriş gecikmeli) **zone**" olarak ilgili program kodlarını girerek tanıtınız.
6. **Keypad** aracılığı ile hırsız alarm santraline kablosuz 1. PIR dedektörünü "normal (giriş gecikmeli) **zone**" olarak ilgili program kodlarını girerek tanıtınız ve sinyal gücünü test ediniz.
7. **Keypad** aracılığı ile hırsız alarm santraline kablosuz 2. PIR dedektörünü "normal (giriş gecikmeli) **zone**" olarak ilgili program kodlarını girerek tanıtınız ve sinyal gücünü test ediniz.
8. Giriş gecikmesini 25 sn., çıkış gecikmesini 15 sn., siren çıkış süresini 5 dk. olacak şekilde ilgili program kodlarını girerek tanıtınız.
9. Hırsız alarm santraline sizin belirleyeceğiniz iki adet kullanıcı şifresi ekleyiniz.
10. Hırsız alarm sistemini ilgili kodları girerek aktif hâle getiriniz ve yaptığınız program kodlarının çalıştığını gözlemleyiniz. Hırsız alarm santralini pasif hâle getiriniz.
11. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.
12. Görsel 5.69'daki bir kablolu dedektörün dirençli ve iki kablosuz dedektörün **zone**larına bağlantı şemasını Görsel 5.68 üzerinde dirençli **zone** bağlantı şemasına dikkat ederek ürününüze uygun olan bağlantı şemasını belirleyip hırsız alarm santrali üzerinde kurunuz. İşlem basamaklarını tekrarlayınız.

Sorular

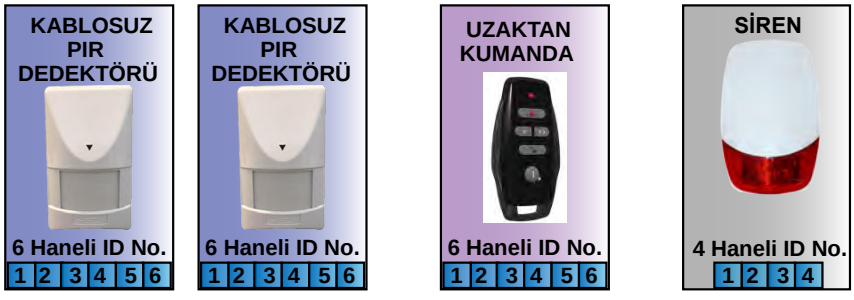
1. Uygulama sırasında kullandığınız hırsız alarm santrallerinin modelini, **zone** bağlantı türünü, direnç değerlerini ve program kodlarını maddeler hâlinde yazınız.
2. Uygulama sırasında kullandığınız sirenin modelini ve tetikleme yöntemini yazınız.
3. Uygulama sırasında kullandığınız siren eğer kablosuz siren olsaydı hırsız alarm santraline tanıtımını nasıl yapardınız? Araştırarak kablolu hırsız alarm santrali üzerinde uygulamasını yapınız.
4. Kablolu ve kablosuz hırsız alarm santralleri uzaktan kumanda ile nasıl kontrol edilebilir? Bu durumu ayrı ayrı hırsız alarm santralleri için araştırınız. Kablolu hırsız alarm santrali üzerinde uygulamasını yapınız.

NOT DEFTERİ

Uygulama Adı: Kablosuz Hırsız Alarm Santralinin Programlanması No.: 12

Amaç: Kablosuz hırsız alarm santralinin programlamasını yapmak.

Kablosuz hırsız alarm santrallerine; kablosuz dedektörler, uzaktan kumanda ürünleri ve kablosuz sirenlerin tanıtımı yapılabilir. Kablosuz dedektörlerin ve uzaktan kumandaların ID numaraları 6 haneli, kablosuz sirenin ID numarası ise 4 hanelidir. Uzaktan kumanda ürünlerinin ve kablosuz sirenin hırsız alarm santraline tanıtılması kablosuz dedektörlerin tanıtılmasına benzerdir. Tercih edilen kablosuz hırsız alarm sistemine göre tanıtma program kodları farklılık gösterebilir.



Görsel 5.70: Kablosuz hırsız alarm sistemi ürünleri

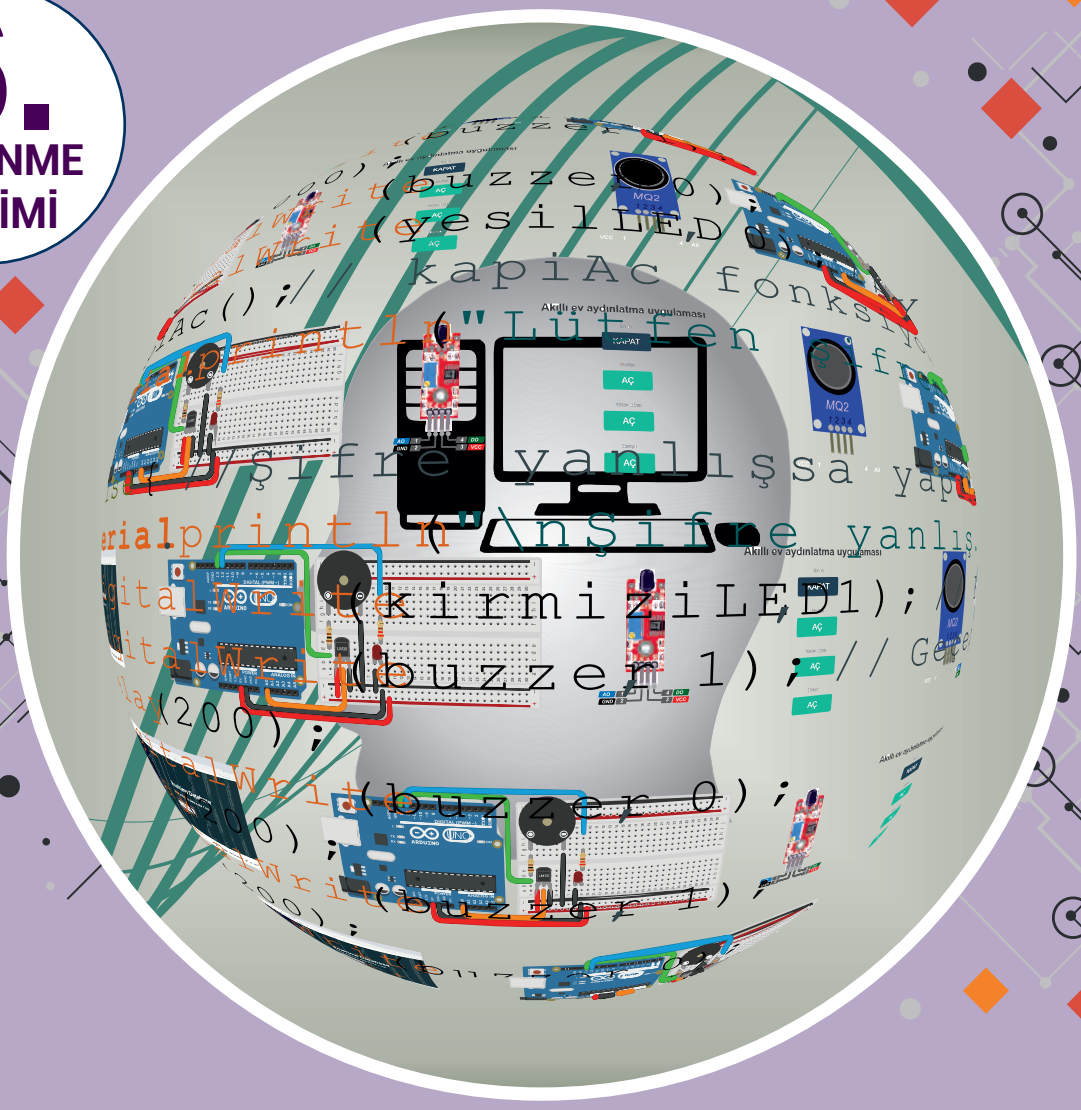
İşlem Basamakları

1. İş güvenliği ve sağlığı kurallarına uyarak hırsız alarm santralinin kapağını açınız.
2. Öğretmeninizin kontrolünde kablosuz hırsız alarm santralinin şehir şebekesi ve akü gerilimlerinin bağlantılarını yapınız.
3. Hırsız alarm santralini resetleme yöntemlerinden biri ile resetleyiniz. **Keypad** aracılığı ile hırsız alarm santralinin tarih ve saat ayarını yapınız.
4. **Keypad** aracılığı ile hırsız alarm santraline kablosuz 1. PIR dedektörünü "normal (giriş gecikmeli) **zone**" olarak ilgili program kodlarını girerek tanıtınız ve sinyal gücünü test ediniz.
5. **Keypad** aracılığı ile hırsız alarm santraline kablosuz 2. PIR dedektörünü "ani **zone**" olarak ilgili program kodlarını girerek tanıtınız ve sinyal gücünü test ediniz.
6. **Keypad** aracılığı ile hırsız alarm santraline kablosuz sireni, program kodlarını girerek tanıtınız ve sinyal gücünü test ediniz.
7. **Keypad** aracılığı ile hırsız alarm santraline uzaktan kumandanızı ilgili program kodlarını girerek tanıtınız.
8. Giriş gecikmesini 25 sn., çıkış gecikmesini 15 sn., siren çıkış süresini 5 dk. olacak şekilde ilgili program kodlarını girerek tanıtınız.
9. Hırsız alarm santraline sizin belirleyeceğiniz iki adet kullanıcı şifresi ekleyiniz.
10. Hırsız alarm sistemini ilgili kodları girerek aktif hâle getiriniz ve yaptığınız program kodlarının çalıştığını gözlemleyiniz. Hırsız alarm santralini pasif hâle getiriniz.
11. Hırsız alarm santralini resetleme yöntemlerinden biriyle resetleyip öğretmeninizin kontrolünde sistemin şebeke gerilimini kesip akü bağlantılarını sökünüz.

Sorular

1. Uygulama sırasında kullandığınız hırsız alarm santrallerinin modelini, **zone** bağlantı türünü ve program kodlarını maddeler hâlinde yazınız.
2. Kablosuz hırsız alarm sistemlerinde kullanılan ürünlerin tanıtımı sırasında sinyal gücü neden ölçülür? Araştırınız.

6. ÖĞRENME BİRİMİ



ELEKTRONİK GÜVENLİK DEVRELERİ

NELER ÖĞRENECEKSİNİZ ?

- Sıcaklık Alev Gaz Dedektörü Devresi
- Hareket Dedektörü Devresi
- Yaklaşım Dedektörü Devresi
- Şifreli Kilit, Kartlı Geçiş Uygulaması
- Parmak İzi Okuma Uygulaması
- Akıllı Ev Aydınlatma Uygulaması

TEMEL KAVRAMLAR

güvenlik
nesnelerin interneti
veri tabanı
parmak izi
akıllı ev
Wi-Fi
şifre
kilit

Uygulama Adı: Sıcaklık Dedektörü Devresi — No.: 1

Amaç: Sıcaklık dedektörü devresi yapmak.

Görsel 6.1'de LM35 sıcaklık dedektörünün pin yapısı görülmektedir. Görsel 6.2'deki LM35'li sıcaklık dedektörü devresi, ortam sıcaklığı 30 °C'yi aşınca ses ve ışıkla ikaz verir.

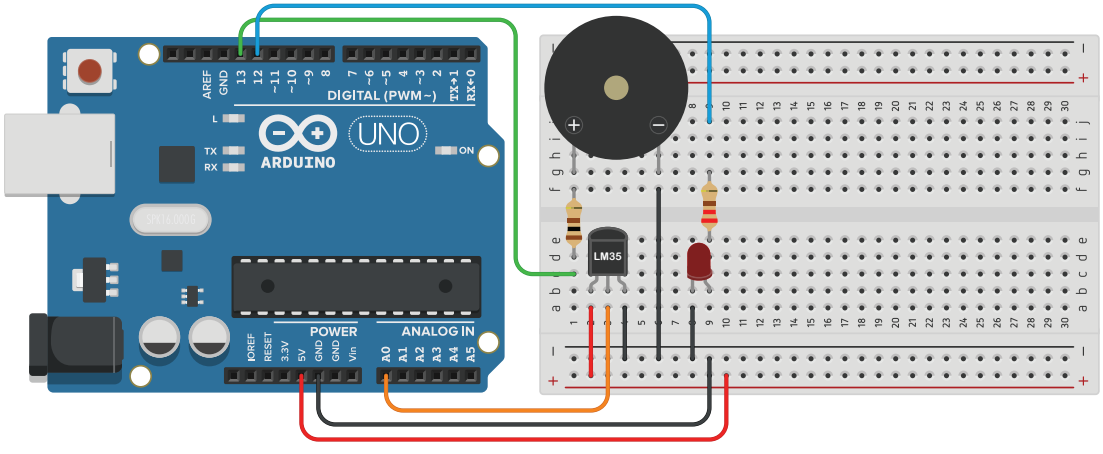


Görsel 6.1: LM35 pin yapısı

+V: 5 V besleme.

OUT: Analog veri çıkışı.

GND: Toprak pini.



Görsel 6.2: Sıcaklık dedektörü devresi

Sıcaklık dedektörü uygulamasının programı aşağıdaki gibidir:

```
const byte buzzer = 13, LED = 12;

void setup() {
  Serial.begin(9600);
  pinMode(buzzer, OUTPUT);
  pinMode(LED, OUTPUT);
}

void loop() {
  int deger = analogRead(A0); // Analog girişi oku.
  float gerilim = (5000.0 / 1023.0) * deger; // Analog girişteki gerilim (mV).
  float  sicaklik = gerilim / 10.0; // Sıcaklık °C
  Serial.print(sicaklik);
```

```
 Serial.println(“ °C”);
  if (sicaklik > 30)
    alarm(); // alarm() fonksiyonunu çağır.
}

void alarm() {
  digitalWrite(buzzer, !digitalRead(buzzer)); // Çıkışı tersle.
  digitalWrite(LED, digitalRead(LED ^ 1)); // Çıkışı tersle.
  delay(1000); // Bir saniye bekle.
}
```

İşlem Basamakları

1. Görsel 6.2'deki devreyi kurunuz. Programı yükleyiniz.

2. LM35'i ısıtarak sıcaklık 30 °C'yi aşınca devrenin verdiği tepkiyi gözlemleyiniz.

SIRA SİZDE

1. Çıkışa LED yerine röle bağlayarak fan çalıştıran uygulamayı gerçekleştiriniz.

2. Aşağıdaki programda hangi sıcaklık sensörü kullanılmıştır? Araştırınız.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>

Adafruit_BME280 bme;

void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.print(“Sıcaklık = “);
  Serial.print(bme.readTemperature());
  Serial.println(“*C”);
  delay(1000);
}
```

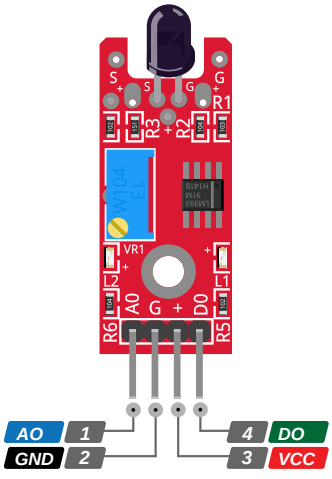
Soru

1. Sıcaklık ölçümü için LM35 yerine hangi sensörler kullanılabilir? Araştırınız.

Uygulama Adı: Alev Dedektörü Devresi — No.: 2

Amaç: Alev dedektörü devresi yapmak.

Alev dedektörleri ortamda alev algıladığında uyarı sinyali verir. Alev dedektörü 760 nm - 1100 nm arasındaki dalga boyuna sahip ışığı (kızılötesi ışınım) tespit etmek için kullanılır. Görsel 6.3'te alev dedektörü pin yapısı, Görsel 6.4'te ise alev dedektörü devresi görülmektedir.



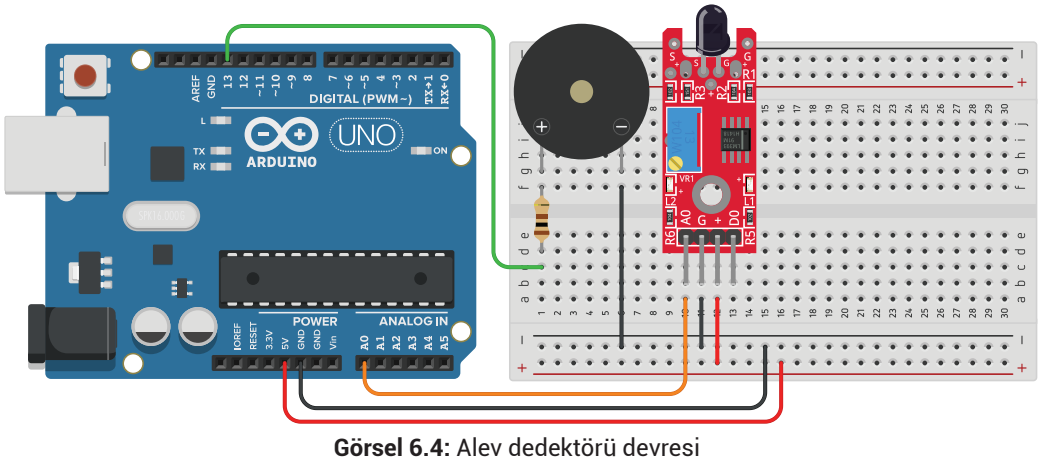
Görsel 6.3: Alev dedektörü pin yapısı

A0: Analog veri çıkışı.

GND: Toprak pini.

VCC: 3,3 V-5 V besleme.

D0: Dijital veri çıkışı.



Görsel 6.4: Alev dedektörü devresi

Alev dedektörü uygulamasının programı aşağıdaki gibidir:

```
const byte buzzer = 13; // Buzzerın takılacağı pin.
void setup() {
  pinMode(buzzer, OUTPUT); // Buzzer pin çıkış olarak ayarlandı.
}
```

```
void loop() {
  int deger = analogRead(A0); // Sensörden analog değer oku.
  if (deger > 300) { // Sensörden okunan değer belirlenen değerinden büyükse çalışır.
    digitalWrite(buzzer, HIGH);
    delay(100);
    digitalWrite(buzzer, LOW);
    delay(100);
  } else { //Sensörden okunan değer belirlenen değerinin altındaysa çalışır.
    digitalWrite(buzzer, LOW);
  }
}
```

İşlem Basamakları

1. Görsel 6.4'teki devreyi kurunuz. Programı yükleyiniz.

2. Sensöre çakmak tutarak sensörün çalışmasını gözlemleyiniz.

SIRA SİZDE

1. Uygulamayı alev değerini LCD ekranda gösterecek şekilde düzenleyiniz.

```
#include <LiquidCrystal.h>
// LCD modülün Arduino pinlerne bağlantısı.
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
  lcd.begin(16, 2); //LCD modülün satır ve sütun sayısı.
}

void loop() {
  int analogGirisDegeri = analogRead(A0); //A0 girişini oku.
  lcd.setCursor(0, 1);
  //İmleci 0. sütun 1. satıra konumlandır.
  lcd.print(analogGirisDegeri); // Okunan analog değerini ekrana yaz.
}
```

Soru

1. Kızılötesi ışınımın elektronik alanındaki kullanım yerlerini araştırınız.

Uygulama Adı: Gaz Dedektörü Devresi — No.: 3

Amaç: Gaz dedektörü devresi yapmak.

Görsel 6.5'te MQ-2 gaz dedektörü pin yapısı görülmektedir. MQ-2 gaz sensörü, 300 ile 10.000 ppm seviyesindeki yanıcı gaz ve sigara dumanını algılar. Görsel 6.6'da görülen gaz sensörü devresi, ortamda gaz algıladığında uyarı sinyalleri verir.



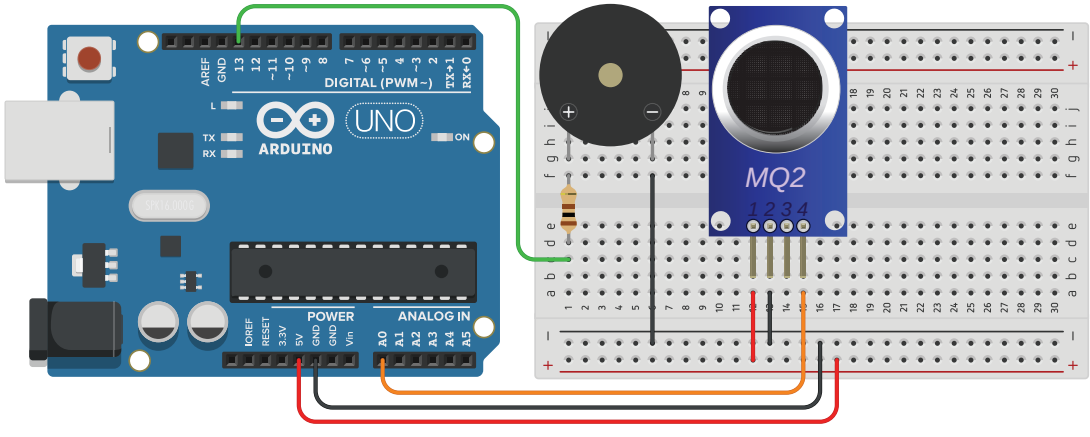
Görsel 6.5: Gaz dedektörü pin yapısı

VCC: 5 V besleme.

GND: Toprak pini.

D0: Dijital veri çıkışı.

A0: Analog veri çıkışı.



Görsel 6.6: Gaz dedektörü devresi

Gaz dedektörü uygulamasının programı aşağıdaki gibidir:

```
const byte buzzer = 13; // Buzzerın takılacağı pin.
void setup() {
  pinMode(buzzer, OUTPUT); // Buzzer pin çıkış olarak ayarlandı.
}
void loop() {
  int deger = analogRead(A0); // Sensörden analog değer oku.
```

```
  if (deger > 300) { // Sensörden okunan değer belirlenen değerinden büyükse çalışır.
    digitalWrite(buzzer, HIGH);
    delay(100);
    digitalWrite(buzzer, LOW);
    delay(100);
  } else { //Sensörden okunan değer belirlenen değerinin altındaysa çalışır.
    digitalWrite(buzzer, LOW);
 }
}
```

İşlem Basamakları

1. Görsel 6.6'daki devreyi kurunuz. Programı yükleyiniz.
2. Dedektöre çakmak gazı tutarak dedektörün çalışmasını gözlemleyiniz.

SIRA SİZDE

1. Uygulamayı gaz değerini LCD ekranda gösterecek şekilde düzenleyiniz.
2. Programı analog giriş değeri 700'ün üzerine çıktığında servo motoru 90° döndüren ve LCD ekranda "Fan aktif" yazacak şekilde düzenleyiniz.

```
#include <LiquidCrystal.h>
#include <Servo.h>
Servo servoMotor;
// LCD modülün Arduino pinlerine bağlantısı.
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup() {
  lcd.begin(16, 2); //LCD modülün satır ve sütun sayısı.
  servoMotor.attach(9);
  pinMode(13, OUTPUT); // Fan bağlı pin.
}
void loop() {
  int analogGirisDegeri = analogRead(A0); //A0 girişini oku.
  lcd.setCursor(0, 1);
  lcd.print(analogGirisDegeri); // Okunan analog değerini ekrana yaz.
  if (analogGirisDegeri > 700) {
    servoMotor.write(90);
    lcd.setCursor(0, 1);
    lcd.print("fan aktif");
    digitalWrite(13, 1); // Fanı çalıştır.
  }
}
```

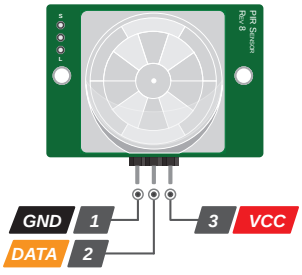
Soru

1. Gaz sensörü çeşitleri nelerdir? Araştırınız.

Uygulama Adı: Hareket Dedektörü Devresi — No.: 4

Amaç: Hareket dedektörü devresi yapmak.

Görsel 6.7'de HC-SR501 hareket dedektörü pin yapısı görülmektedir. Hareket dedektörüne LDR eklendiğinde HC-SR501 yalnızca karanlıkta çalışacak şekilde ayarlanabilir. Görsel 6.8'deki hareket dedektörü devresi ortamda canlı algıladığında LED'i yakmakta ve siren çalmaktadır.

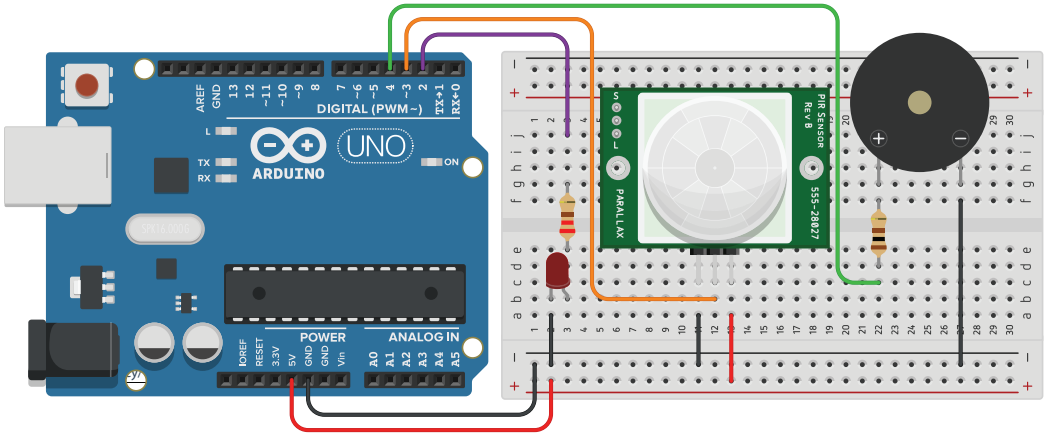


Görsel 6.7: Hareket dedektörü pin yapısı

GND: Topraklama pini.

DATA: Dijital veri çıkışı (3,3 V).

VCC: 5 V - 12 V besleme.



Görsel 6.8: Hareket dedektörü devresi

Hareket dedektörü uygulamasının programı aşağıdaki gibidir:

```
const byte LED = 2, PIR = 3, buzzer = 4; // Pin bağlantıları.
void setup() {
  pinMode(LED, OUTPUT);
  pinMode(PIR, INPUT);
  pinMode(buzzer, OUTPUT);
  Serial.begin(9600);
}
void loop() {
  if (digitalRead(PIR))  { //PIR çıkış veriyorsa...
    digitalWrite(LED, HIGH); // LED'i yak.
```

```
    Serial.println("Hareket algılandı.");
    siren(); // Siren fonksiyonunu çağır.
  }
  else  { //PIR çıkış vermiyorsa...
    digitalWrite(LED, LOW); // LED'i söndür.
    Serial.println("Hareket algılanmadı.");
  }
}
void siren() {
  for (int frekans = 500; frekans < 2000; frekans += 10) { // Frekası artır.
    tone(buzzer, frekans);
    delay(10);
  }
  for (int frekans = 2000; frekans > 500; frekans -= 10) { // Frekansı azalt.
    tone(buzzer, frekans);
    delay(10);
  }
}
```

İşlem Basamakları

1. Görsel 6.8'deki devreyi kurunuz. Programı yükleyiniz.

2. Hareketsiz bir ortamda HC-SR501 dedektörünün önüne geçerek çalışmasını gözlemleyiniz.

SIRA SİZDE

1. Uygulamayı hareket algılandığında servo motor çalıştıracak şekilde düzenleyiniz.

```
#include <Servo.h>
Servo servo; // Servo isimli nesne oluşturuldu.
const byte PIR = 3;
void setup() {
  servo.attach(9); // Servo motor 9 numaralı PWM çıkışına bağlı.
  pinMode(PIR, INPUT);
}
void loop() {
  if (digitalRead(PIR))  { //PIR çıkış veriyorsa...
    servo.write(90); // Servo motorun gideceği derece.
    delay(15);
  }
  servo.write(0); // Servo motorun gideceği derece.
  delay(15);
}
```

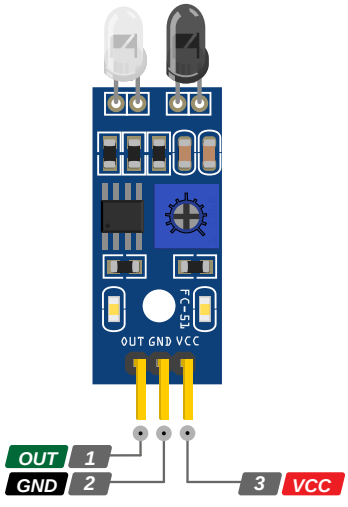
Soru

1. Sıcaklık ve kütle, kızılötesi ışınımı ve PIR dedektörünün çalışmasını nasıl etkiler? Araştırınız.

Uygulama Adı: Yaklaşım Dedektörü Devresi — No.: 5

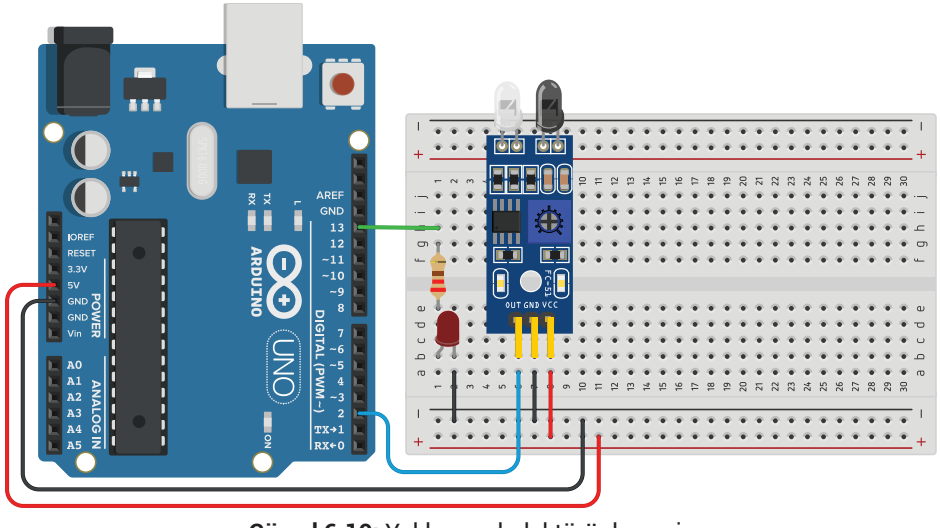
Amaç: Yaklaşım dedektörü devresi yapmak.

Görsel 6.9'da yaklaşım dedektörü modülü ve pin yapısı görülmektedir. Dedektör modülü bir çift IR alıcı verici LED'e sahiptir. Verici LED kızılötesi frekansta ışık verir. 2 cm-35 cm aralığında bulunan engelin yüzeyine çarpıp yansıyan ışık alıcı sensör tarafından okunur. Dijital çıkış veren modül üzerindeki yeşil ışık yanar ve dijital çıkıştan da **LOW** (lojik 0) okunur. Algılama uzaklığı potansiyometre ile ayarlanır. Siyah yüzey en düşük algılama mesafesini sağlarken beyaz yüzeyse en uzak algılama mesafesine izin vermektedir. Objenin büyüklüğü, konumu, dedektöre göre açısı algılamayı etkileyen diğer değişkenlerdir. Görsel 6.10'daki devre yaklaşım dedektörüne bir cisim yaklaştığında LED'le uyarı verir.



Görsel 6.9: Yaklaşım dedektörü modülü ve pin yapısı

OUT: Dijital veri çıkışı.
GND: Toprak pini.
VCC: 5 V besleme.



Görsel 6.10: Yaklaşım dedektörü devresi

Yaklaşım dedektörü uygulamasının programı aşağıdaki gibidir:

```
const byte IR = 2, LED = 13;
void setup() {
  pinMode(IR, INPUT);
  pinMode(LED, OUTPUT);
}
void loop() {
  bool IRdurum = digitalRead(IR); // Cisim yok lojik 1, cisim var lojik 0.
  if (IRdurum == 0) // Cisim algılandıysa...
    digitalWrite(LED, HIGH);
  else digitalWrite(LED, LOW);
}
```

İşlem Basamakları

1. Görsel 6.10'daki devreyi kurunuz. Programı yükleyiniz.

2. IR alıcı verici sensöre bir cisim yaklaştırarak sensörün çalışmasını gözlemleyiniz.

SIRA SİZDE

1. Uygulamayı cisim algılandığında servo motor çalıştıracak şekilde düzenleyiniz.

```
#include <Servo.h>
Servo servo; // Servo isimli nesne oluşturuldu.
const byte IR = 2, LED = 13;
void setup() {
  servo.attach(9); // Servo motor 9 numaralı PWM çıkışına bağlı.
  pinMode(IR, INPUT);
}
void loop() {
  bool IRdurum = digitalRead(IR); // Cisim yok lojik 1, cisim var lojik 0.
  if (IRdurum == 0) { // Cisim algılandıysa...
    servo.write(90); // Servo motorun gideceği derece.
    delay(15);
  }
  else {
    servo.write(0); // Servo motorun gideceği derece.
    delay(15);
  }
}
```

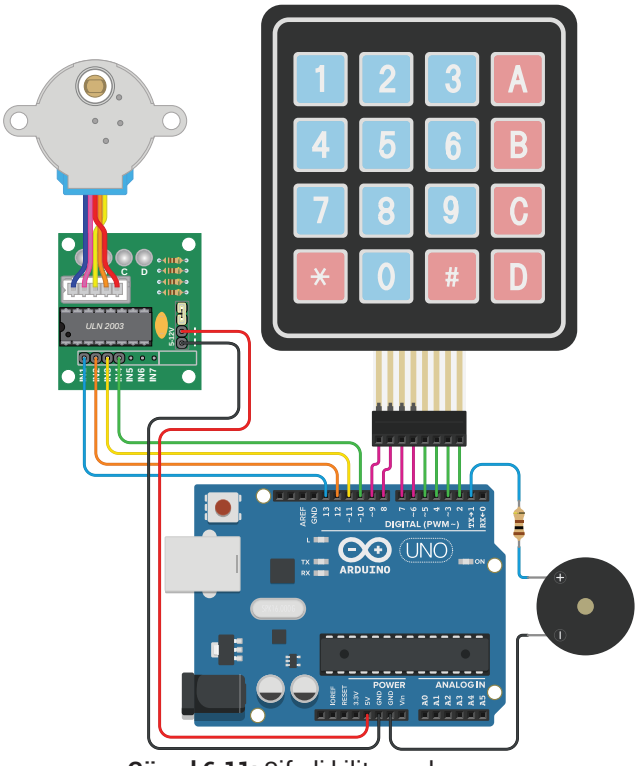
Soru

1. Yaklaşım dedektörü ile hareket dedektörünün (PIR) çalışmasındaki farklar nelerdir? Açıklayınız.

Uygulama Adı: Şifreli Kilit Uygulaması — No.: 6

Amaç: Şifreli kilit uygulaması yapmak.

Görsel 6.11'deki uygulamada tuş takımından doğru şifre girildiğinde step motor 90 °C'lik dönüş yapar. Geçerli ve geçersiz şifreler için farklı ses sinyalleri verir.



Görsel 6.11: Şifreli kilit uygulaması

Şifreli kilit uygulamasının programı aşağıdaki gibidir:

```
#include <Keypad.h> // Tuş takımı kütüphanesi. V3.1.1

byte hane = 0;
char sifre[] = "123ABC"; // Kullanılacak şifre.
char giris[sizeof(sifre)]; //Şifre uzunluğu kadar dizi oluştur.
const byte buzzer = 1; // Buzzer pini.

//---Tuş takımı ayarları.---
const byte satir = 4; //Dört satır.
const byte sutun = 4; //Dört sütün.
char tusDizilimi[satir][sutun] = {//Tuş takımındaki sembollerin dizilimini tanımlama.
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
byte satirPinleri[satir] = {9, 8, 7, 6};
//{R1, R2, R3, R4} Satır pinlerinin Arduino pinlerine bağlantısı.
byte sutunPinleri[sutun] = {5, 4, 3, 2};
//{C1, C2, C3, C4} Sütun pinlerinin Arduino pinlerine bağlantısı.

//tusTakimi adında nesne oluşturma.
Keypad tusTakimi = Keypad( makeKeymap(tusDizilimi), satirPinleri, sutunPinleri, satir,
sutun);
```

```
void setup() {
  pinMode(buzzer, OUTPUT); // Buzzer.
  pinMode(10, OUTPUT); // Step motor pinleri.
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);

  Serial.begin(9600);
  Serial.println("Lütfen şifrenizi giriniz: ");
}

void loop() {
  char tus = tusTakimi.waitForKey(); //Tuşa basılana kadar bekle.
  Serial.print(tus); //Basılan tuşu ekrana yazdır.
  giris[hane] = tus; //Sırayla her tuşu giriş dizisine yükle.
  hane++;

  if (hane == sizeof(sifre) - 1) { //Şifre uzunluğu kadar tuşa basıldığında...
    if (!strcmp(giris, sifre)) { //İki diziyi karşılaştır. Şifre doğruysa yapılacaklar...
      Serial.println("\nŞifre doğru. Kapı açıldı");
      digitalWrite(buzzer, 1); // Buzzer bir saniye çalar.
      delay(1000);
      digitalWrite(buzzer, 0);
      stepMotor(); // Step motor çalıştıran fonksiyonu çağır.
      Serial.println("Lütfen şifrenizi giriniz: ");
    }
    else { //Şifre yanlışsa yapılacaklar...
      Serial.println("\nŞifre yanlış. Lütfen tekrar deneyiniz.");
      digitalWrite(buzzer, 1); // Geçersiz kart için buzzer sesi.
      delay(200);
      digitalWrite(buzzer, 0);
      delay(200);
      digitalWrite(buzzer, 1);
      delay(200);
      digitalWrite(buzzer, 0);
    }
    hane = 0;
  }
}

void stepMotor() { // Step motor Yaklaşık 90° döner.
  int x, adimSayisi = 500, sure = 2;

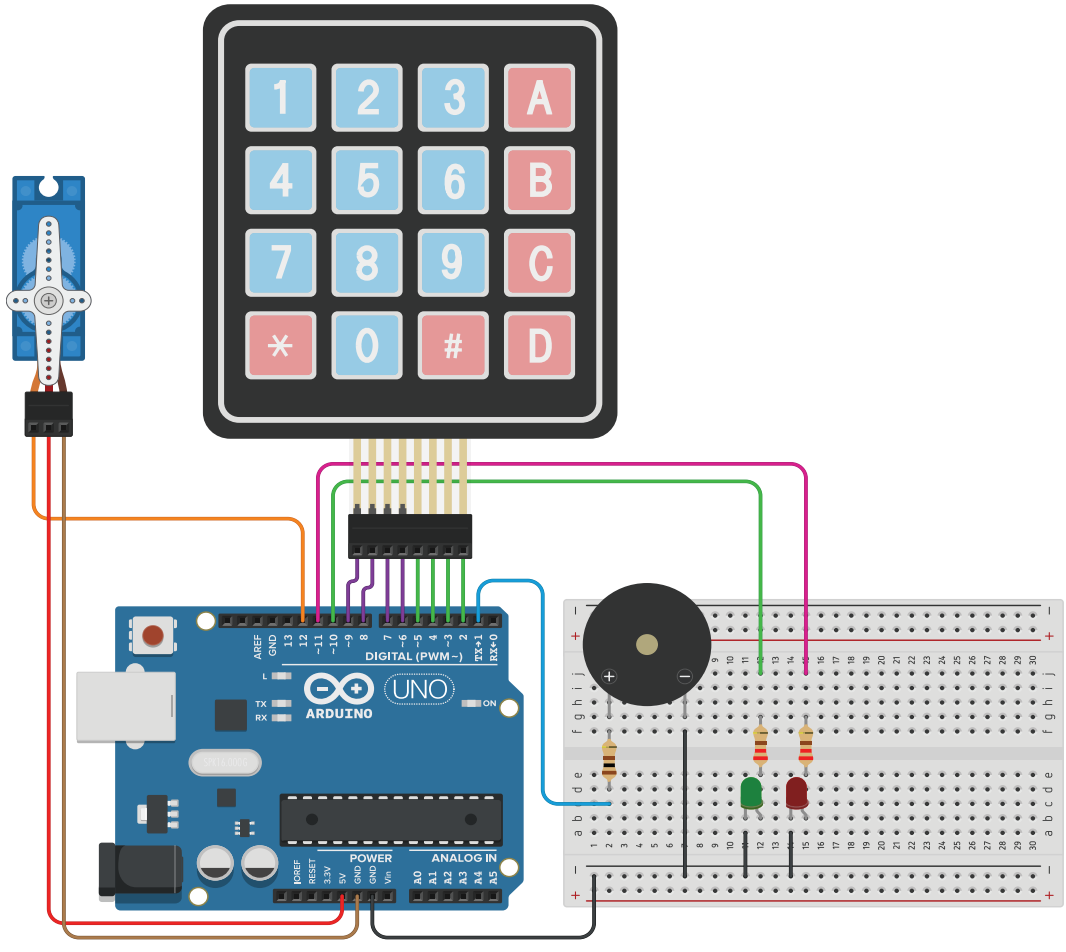
  for (x = 0; x < adimSayisi / 4; x++) {
    for (byte i = 10; i <= 13; i++) { //Sağa dön.
      digitalWrite(i, 1);
      delay(sure);
      digitalWrite(i, 0);
    }
  }
  delay(1000); // Kapı açık kalma süresi.
  for (x = 0; x < adimSayisi / 4; x++) {
    for (byte i = 13; i >= 10; i--) { //Sola dön.
      digitalWrite(i, 1);
      delay(sure);
      digitalWrite(i, 0);
    }
  }
}
```

İşlem Basamakları

1. Görsel 6.11'deki devreyi kurunuz.
2. Taslaktaki tuş takımı dizilimini elinizdeki tuş takımına göre düzenleyiniz.
3. Programı yükleyiniz.
4. Basılan tuşu seri port ekranında gözlemleyiniz.

SIRA SİZDE

1. Doğru şifre girdiğinizde servo motor kontrol eden uygulamayı gerçekleştiriniz.




```
#include <Keypad.h> // Tuş takımı kütüphanesi. V3.1.1
#include <Servo.h> // Dahili servo kütüphanesi v1.1.8

Servo servo; // Servo isimli nesne oluşturuldu.

const byte buzzer = 1, yesilLED = 10, kirmiziLED = 11;
byte hane = 0;
char sifre[] = "123ABC"; // Kullanılacak şifre.
char giris[sizeof(sifre)]; //Şifre uzunluğu kadar dizi oluştur.

//---Tuş takımı ayarları.---
const byte satir = 4; //Dört satır.
const byte sutun = 4; //Dört sütun.
char tusDizilimi[satir][sütun] = {//Tuş takımındaki sembollerin dizilimini tanımlama.
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
byte satirPinleri[satir] = {9, 8, 7, 6};
//{R1, R2, R3, R4} Satır pinlerinin Arduino pinlerine bağlantısı.
byte sutunPinleri[sutun] = {5, 4, 3, 2};
//{C1, C2, C3, C4} Sütun pinlerinin Arduino pinlerine bağlantısı.

//tusTakimi adında nesne oluşturma.
Keypad tusTakimi = Keypad( makeKeymap(tusDizilimi), satirPinleri, sutunPinleri, satir, sutun);

void setup() {
  pinMode(buzzer, OUTPUT);
  pinMode(yesilLED, OUTPUT);
```

```
  pinMode(kirmiziLED, OUTPUT);
  servo.attach(12); // Servo motor 12 numaralı PWM çıkışına bağlı.
  Serial.begin(9600);
  Serial.println("Lütfen şifrenizi giriniz: ");
}

void loop() {
  char tus = tusTakimi.waitForKey(); //Tuşa basılana kadar bekle.
  Serial.print(tus); //Basılan tuşu ekrana yazdır.
  giris[hane] = tus; //Sırayla her tuşu giris dizisine yükle.
  hane++;

  if (hane == sizeof(sifre) - 1) { //Şifre uzunluğu kadar tuşa basıldığında...
    if (!strcmp(giris, sifre)) { //İki diziyi karşılaştır. Şifre doğruysa yapılacaklar...
      Serial.println("\nŞifre doğru. Kapı açıldı");
      digitalWrite(yesilLED, 1); // Yeşil LED'i yak.
      digitalWrite(buzzer, 1); // Buzzer bir saniye çalar.
      delay(1000);
      digitalWrite(buzzer, 0);
      digitalWrite(yesilLED, 0); // Yeşil LED'i söndür.
      kapiAc(); // kapiAc fonksiyonunu çağır.
      Serial.println("Lütfen şifrenizi giriniz: ");
    }
   else { //Şifre yanlışsa yapılacaklar...
      Serial.println("\nŞifre yanlış. Lütfen tekrar deneyiniz.");
      digitalWrite(kirmiziLED, 1); // Kırmızı LED'i yak.
      digitalWrite(buzzer, 1); // Geçersiz şifre için buzzer sesi.
      delay(200);
      digitalWrite(buzzer, 0);
      delay(200);
      digitalWrite(buzzer, 1);
      delay(200);
      digitalWrite(buzzer, 0);
      digitalWrite(kirmiziLED, 0); // Kırmızı LED'i söndür.
    }
    hane = 0;
  }
}

void kapiAc() {
  servo.write(90); //Servo motoru 180 dereceye getirir.
  delay(3000);
  servo.write(0); //Servo motoru 0 dereceye getirir.
  delay(1000);
}
```

Sorular

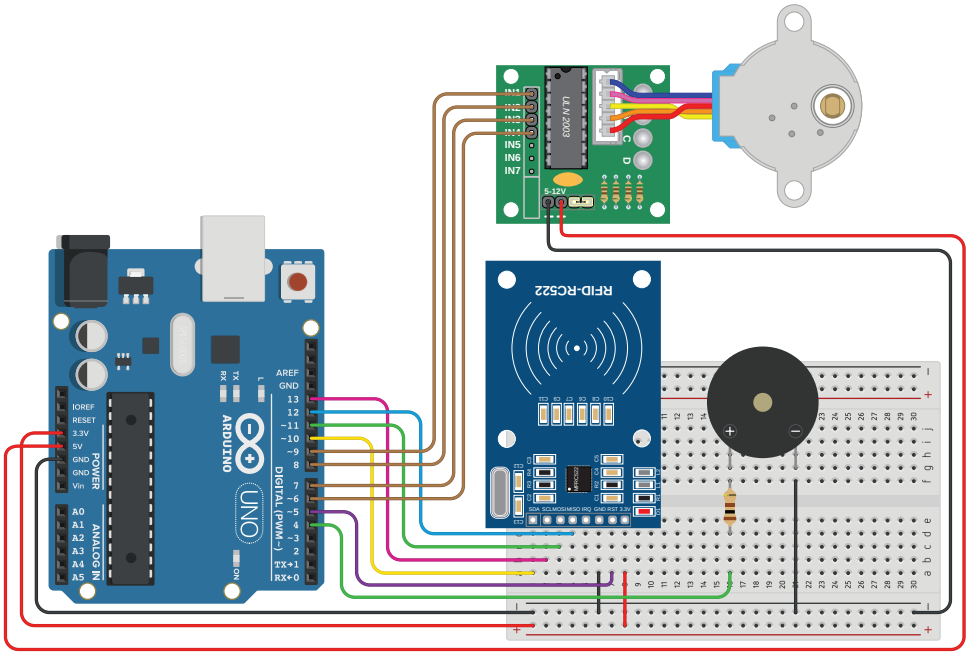
1. Tuş takımında "1" rakamına basıldığında farklı bir karakter yazdırılabilir mi? Nedenini açıklayınız.

2. Kütüphane ile fonksiyon arasındaki fark nedir? Açıklayınız.

Uygulama Adı: Kartlı Geçiş Uygulaması — No.: 7

Amaç: Kartlı geçiş uygulaması yapmak.

Görsel 6.12'deki uygulamada RFID okuyucuya geçerli bir kart okutulduğunda step motor 90 °C'lik dönüş yapar. Geçerli ve geçersiz kartlar için farklı ses sinyalleri verir. Geçersiz kartın ID'si seri ekrandan okunarak istenirse geçerli kart listesine eklenir.



Görsel 6.12: RFID uygulaması

Kartlı geçiş uygulamasının programı aşağıdaki gibidir:

```
#include <SPI.h> // Dahili SPI kütüphanesi
#include <MFRC522.h>  // v1.4.9
#define reset 9 // Reset pini
#define ss 10  // SS pin
MFRC522 kartOkuyucu(ss, reset);  // Kart Okuyucu isimli nesne oluşturuldu.
byte i, j, k;
byte kartSayisi = 2; //Kart sayısı doğru girilmelidir.
byte kartlar[][4] = {
  {0xA6, 0x06, 0x1E, 0xF9}, // İlk kartın dört bytelık ID'si HEX olarak tanımlandı.
  {0xA6, 0x06, 0x1E, 0xF8}, // İkinci kart...
};
void setup() {
  Serial.begin(9600);                    //Bilgisayarın seri ekranıyla UART iletişimi.
  SPI.begin();                           // MFRC522 kart okuyucusuyla SPI iletişimi.
```

```
  kartOkuyucu.PCD_Init(); // MFRC522'yi başlat
  Serial.println("Lütfen kartı okutunuz.");
  pinMode(4, OUTPUT); // Buzzer.
  pinMode(6, OUTPUT); // Step motor pinleri.
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
}
void loop() {
  if ( ! kartOkuyucu.PICC_IsNewCardPresent())  //Bir kart okunana kadar bekle.
    return;
  if ( ! kartOkuyucu.PICC_ReadCardSerial())  // Kartı oku.
    return;
  if (kartKontrol()) { // Kart geçerliyse yapılacaklar...
    Serial.println("Kart geçerli.");
    digitalWrite(4, 1); // Buzzer bir saniye çalar.
    delay(1000);
    digitalWrite(4, 0);
    stepMotor(); // Step motor çalıştıran fonksiyonu çağır.
  }
  else { // Kart geçerli değilse yapılacaklar...
    for (i = 0; i < 4; i++) {
      Serial.print(kartOkuyucu.uid.uidByte[i], HEX); //Geçersiz kartın ID'sini ekrana yazdır.
      Serial.print(" ");
    }
    Serial.println(" numaralı kartınız sisteme tanımlı değil!");
    digitalWrite(4, 1); // Geçersiz kart için buzzer sesi.
    delay(200);
    digitalWrite(4, 0);
    delay(200);
    digitalWrite(4, 1);
    delay(200);
    digitalWrite(4, 0);
  }
  delay(500);
  Serial.println("\n Lütfen kartı okutunuz.");
}
```

```
boolean kartKontrol() { // Kart geçerliye 1, değilse 0 döndürür.
  k = 0;
  for (j = 0; j < kartSayisi; j++) { // Kayıtlı tüm kartlara bak.
    for (i = 0; i < 4; i++) { // Her kartın ID'si dört byte.
      if (kartlar[j][i] == kartOkuyucu.uid.uidByte[i]) {//Okunan kartı kayıtlı olanlarla karşılaştır.
        k++;
        if (k == 4) { // Tüm bytelar doğrulandıysa...
          return 1; // 1 döndür.
        }
      }
    }
  }
  return 0; // Okunan kart kayıtlı kartlarla eşleşmediyse 0 döndür.
}
void stepMotor() { // Step motor Yaklaşık 90° döner.
  int x, adimSayisi = 500, sure = 2;

  for (x = 0; x < adimSayisi / 4; x++) {
    for (byte i = 6; i <= 9; i++) { //Sağa dön.
      digitalWrite(i, 1);
      delay(sure);
      digitalWrite(i, 0);
    }
  }
  delay(1000); // Kapı açık kalma süresi.
  for (x = 0; x < adimSayisi / 4; x++) {
    for (byte i = 9; i >= 6; i--) { //Sola dön.
      digitalWrite(i, 1);
      delay(sure);
      digitalWrite(i, 0);
    }
  }
}
```

İşlem Basamakları

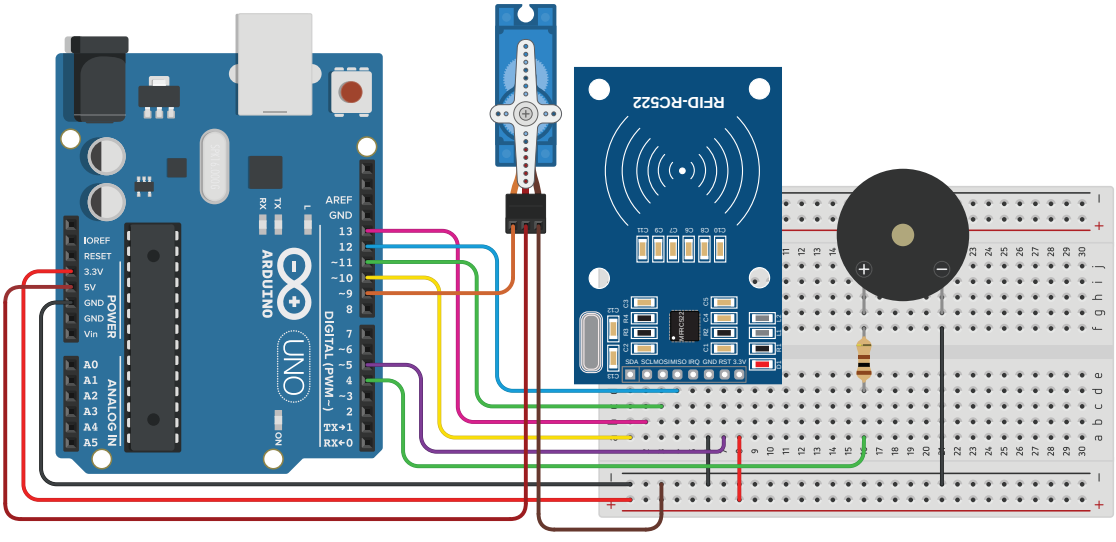
1. Görsel 6.12'deki devreyi kurunuz. Programı yükleyiniz.
2. Seri port ekranını açınız.
3. Kartı okutunuz. Kartın ID'si **hex.** formatında ekranda yazacaktır.
4. Kartın **hex**. kodlarını programdakiyle değiştiriniz.
5. Step motorun hareketinin gözlemlenebilmesi için miline çubuk benzeri bir nesne bağlayınız.
6. Geçerli kartı okutarak step motorun çalışmasını gözlemleyiniz.

DİKKAT

ID'deki **byte**lar seri ekrana yazdırılırken 16'dan küçük sayılar tek haneli yazılır. Örneğin **byte**ın değeri 06'ysa 6 şeklinde yazar. ID, her **byte** arasına boşluk bırakılmadan yazdırılırsa yanlış tanımlanabilir.

SIRA SİZDE

1. Kart bilgilerini EPPROM hafızada tutan ve çıkışında servo motor kontrol eden uygulamayı yapınız.



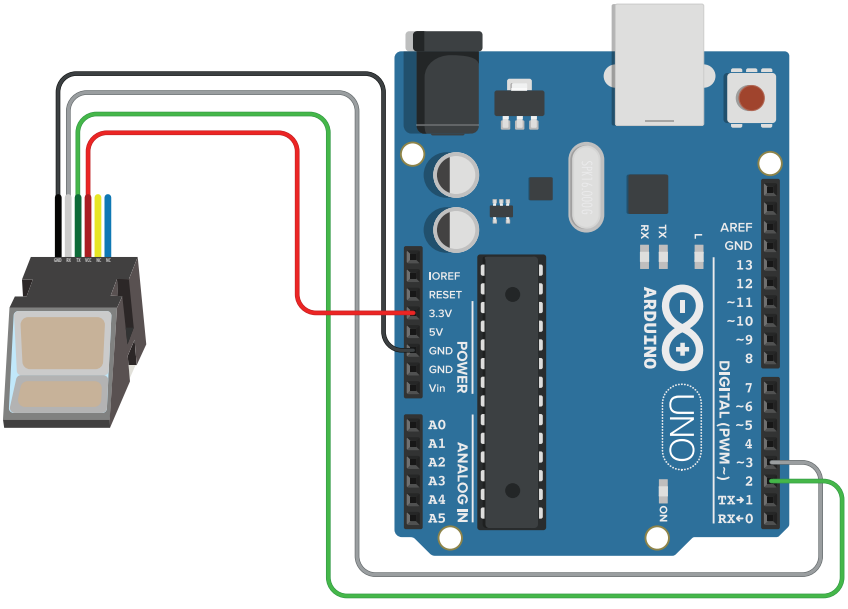
Soru

1. Kartlı geçiş sisteminde akıllı telefon kart işlevini görebilir mi? Araştırınız.

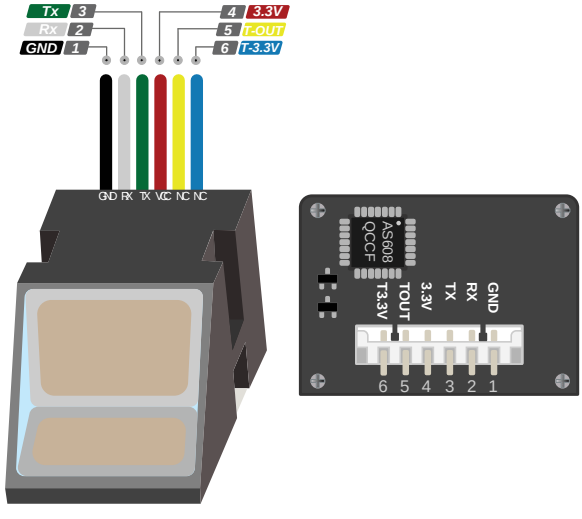
Uygulama Adı: Parmak İzi Okuma Uygulaması — No.: 8

Amaç: Parmak izi okuma uygulaması yapmak.

Görsel 6.13 a'da parmak izi okuma uygulaması görülmektedir. UART haberleşme protokolü kullanılan sensör Arduino'da **SoftwareSerial** kütüphanesi kullanılarak 2 ve 3 numaralı pinlere bağlanmıştır. Görsel 6.13b'de parmak izi sensörünün pin yapısı verilmiştir. Parmak izi kaydetme programıyla kaydedilen parmak izleri, parmak izi okuma programıyla test edilir. İşlemler seri ekran kullanılarak yapılır.



Görsel 6.13a: Parmak izi okuma uygulaması



Görsel 6.13b: Parmak izi sensörünün pin yapısı

VCC: 3,3 V-5 V besleme.

GND: Topraklama pini.

TX: Veri çıkışı (3,3 V).

RX: Veri girişi (3,3 V).

Parmak izi kaydetme uygulamasının programı aşağıdaki gibidir:

```
#include <Adafruit_Fingerprint.h> // v2.1.0

SoftwareSerial seriIletisim(2, 3); // Arduino'da 2. pini Rx, 3. pini TX olarak kullan.

Adafruit_Fingerprint parmakIzi = Adafruit_Fingerprint(&seriIletisim);

byte id;

void setup() {
  Serial.begin(9600);
  Serial.println("\n\nParmak izi kaydetme.");

  parmakIzi.begin(57600); // Sensör seri portu.

  if ( parmakIzi.verifyPassword()) {
    Serial.println("Parmak izi sensörü bulundu.");
  } else {
    Serial.println("Parmak izi sensörü bulunamadı.(");
    while (1) {
      delay(1);
    }
  }

  Serial.println(F("Parmak izi sensörü parametreleri: "));
  parmakIzi.getParameters();
  Serial.print(F("Durum: 0x")); Serial.println( parmakIzi.status_reg, HEX);
  Serial.print(F("Sistem ID: 0x")); Serial.println( parmakIzi.system_id, HEX);
  Serial.print(F("Kapasite: ")); Serial.println( parmakIzi.capacity);
  Serial.print(F("Güvenlik seviyesi: ")); Serial.println( parmakIzi.security_level);
  Serial.print(F("Cihaz adresi: ")); Serial.println( parmakIzi.device_addr, HEX);
  Serial.print(F("Packet len: ")); Serial.println( parmakIzi.packet_len);
  Serial.print(F("Baud rate: ")); Serial.println( parmakIzi.baud_rate);
}

byte numaraOku(void) {
  byte num = 0;

  while (num == 0) {
    while (! Serial.available());
    num = Serial.parseInt();
  }
  return num;
}

void loop() {
  Serial.println("1-127 arası ID kayıt numarası giriniz: ");
  id = numaraOku();
  if (id == 0) { // 0 olmaz.
    return;

   }
  Serial.print("Kaydedilen ID: ");
  Serial.println(id);

  while (!  parmakIziKaydet() );
}

byte parmakIziKaydet() {

  int p = -1;
Serial.print("Kayıt ID'si: "); Serial.println(id);
  while (p != FINGERPRINT_OK) {
    p = parmakIzi.getImage();
    switch (p) {
case FINGERPRINT_OK:
        Serial.println("\nGörüntü alındı");
        break;
 case FINGERPRINT_NOFINGER:
```

```
Serial.print(".");
        delay(200);
        break;
      case FINGERPRINT_PACKETRECIEVEERR:
        Serial.println("İletişim hatası.");
        break;
      case FINGERPRINT_IMAGEFAIL:
        Serial.println("Görüntüleme hatası.");
        break;
      default:
        Serial.println("Bilinmeyen hata.");
        break;
    }
  }

  p = parmakIzi.image2Tz(1);
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Görüntü işlendi.");
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Görüntü karışık.");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("İletişim hatası.");
      return p;
    case FINGERPRINT_FEATUREFAIL:
      Serial.println("Parmak izi özellikleri bulunamadı.");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
      Serial.println("Parmak izi özellikleri bulunamadı.");
      return p;
    default:
      Serial.println("Bilinmeyen hata.");
      return p;
  }

  Serial.println("Parmağınızı kaldırınız.");
  delay(2000);
  p = 0;
  while (p != FINGERPRINT_NOFINGER) {
    p = parmakIzi.getImage();
  }
  Serial.print("ID "); Serial.println(id);
  p = -1;
  Serial.println("Aynı parmağı tekrar yerleştiriniz.");
  while (p != FINGERPRINT_OK) {
    p = parmakIzi.getImage();
    switch (p) {
      case FINGERPRINT_OK:
        Serial.println("\nGörüntü alındı.");
        break;
      case FINGERPRINT_NOFINGER:
        Serial.print(".");
        delay(200);
        break;
   case FINGERPRINT_PACKETRECIEVEERR:
        Serial.println("İletişim hatası.");
        break;
      case FINGERPRINT_IMAGEFAIL:
        Serial.println("Görüntü hatası.");
        break;
      default:
        Serial.println("Bilinmeyen hata.");
        break;
    }
  }
p = parmakIzi.image2Tz(2);
  switch (p) {
    case FINGERPRINT_OK:
```

```
Serial.println("Görüntü oluşturuldu.");
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Görüntü karışık.");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("İletişim hatası.");
      return p;
    case FINGERPRINT_FEATUREFAIL:
      Serial.println("Parmak izi özellikleri bulunamadı.");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
      Serial.println("Parmak izi özellikleri bulunamadı.");
      return p;
    default:
      Serial.println("Bilinmeyen hata.");
      return p;
  }

  Serial.print("Kayıt numarası: ");  Serial.println(id);

  p = parmakIzi.createModel();
  if (p == FINGERPRINT_OK) {
    Serial.println("Parmak izi eşleşti.");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("İletişim hatası.");
    return p;
  } else if (p == FINGERPRINT_ENROLLMISMATCH) {
    Serial.println("Parmak izi eşleşmedi.");
    return p;
  } else {
    Serial.println("Bilinmeyen hata.");
    return p;
  }

  Serial.print("ID "); Serial.println(id);
  p = parmakIzi.storeModel(id);
  if (p == FINGERPRINT_OK) {
    Serial.println("Kaydedildi!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("İletişim hatası.");
    return p;
  } else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Bu alana kaydedilemedi.");
    return p;
  } else if (p == FINGERPRINT_FLASHERR) {
    Serial.println("Yazma hatası.");
    return p;
   } else {
  Serial.println("Bilinmeyen hata.");
    return p;
  }
  return true;
}
```

Parmak izi okuma uygulamasının programı aşağıdaki gibidir:

```
#include <Adafruit_Fingerprint.h> // v2.1.0

SoftwareSerial seriIletisim(2, 3); // Arduino'da 2. pini Rx, 3. pini TX olarak kullan.

Adafruit_Fingerprint parmakIzi = Adafruit_Fingerprint(&seriIletisim);

void setup() {
  Serial.begin(9600);
  Serial.println("\n\nParmak izi okuma.");
  parmakIzi.begin(57600); // Sensör seri portu.

  if (parmakIzi.verifyPassword()) {
```

```
Serial.println("Parmak izi sensörü tespit edildi.");
  } else {
    Serial.println("Parmak izi sensörü tespit edilemedi :(");
    while (1) {
      delay(1);
    }
  }

  parmakIzi.getTemplateCount();
  Serial.print("Hafizada "); Serial.print(parmakIzi.templateCount); Serial.println(" parmak izi mevcut.");
  Serial.println("Lütfen parmağınızı okuyucuya yerleştiriniz.");
  Serial.println("Parmak izi tespit ediliyor...");
}

void loop() {
  getFingerprintIDez();
  delay(50);
}

byte getFingerprintID() {
  byte p = parmakIzi.getImage();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Görüntü kaydedildi.");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println("Parmak izi yok");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("İletişim hatası.");
      return p;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Görüntüleme hatası.");
      return p;
    default:
      Serial.println("Bilinmeyen hata.");
      return p;
  }

  p = parmakIzi.image2Tz();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Görüntü işlendi.");
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Görüntü karışık.");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("İletişim hatası.");
      return p;
    case FINGERPRINT_FEATUREFAIL:
      Serial.println("Parmak izi özellikleri bulunamadı.");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
      Serial.println("Parmak izi özellikleri bulunamadı.");
      return p;
    default:
      Serial.println("Bilinmeyen hata.");
      return p;
  }

  p = parmakIzi.fingerFastSearch();
  if (p == FINGERPRINT_OK) {
    Serial.println("Bir parmak izi ile eşleşti!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("İletişim hatası.");
    return p;
  } else if (p == FINGERPRINT_NOTFOUND) {
    Serial.println("Eşleşme bulunamadı.");
    return p;
  } else {
    Serial.println("Bilinmeyen hata.");
```

```
    return p;
  }

  Serial.print("Bulunan ID: "); Serial.print(parmakIzi.fingerID);
  Serial.print(" doğruluk seviyesi: "); Serial.println(parmakIzi.confidence);

  return parmakIzi.fingerID;
}

int getFingerprintIDez() {
  byte p = parmakIzi.getImage();
  if (p != FINGERPRINT_OK) return -1;

  p = parmakIzi.image2Tz();
  if (p != FINGERPRINT_OK) return -1;

  p = parmakIzi.fingerFastSearch();
  if (p != FINGERPRINT_OK) return -1;

  Serial.print("Bulunan ID: "); Serial.print(parmakIzi.fingerID);
  Serial.print(" doğruluk seviyesi: "); Serial.println(parmakIzi.confidence);
  return parmakIzi.fingerID;
}
```

Parmak izi silme uygulamasının programı aşağıdaki gibidir:

```
#include <Adafruit_Fingerprint.h> // v2.1.0

SoftwareSerial seriIletisim(2, 3); // Arduino'da 2. pini Rx, 3. pini TX olarak kullan.

Adafruit_Fingerprint parmakIzi = Adafruit_Fingerprint(&seriIletisim);

void setup() {
  Serial.begin(9600);
  Serial.println("\n\nParmak izi silme.");

  parmakIzi.begin(57600); // Sensör seri portu.
 if ( parmakIzi.verifyPassword()) {
    Serial.println("Parmak izi sensörü tespit edildi.");
  } else {
    Serial.println("Parmak izi sensörü tespit edilemedi :(");
    while (1) {
      delay(1);
    }
  }
}

byte numaraOku(void) {
  byte num = 0;

  while (num == 0) {
    while (! Serial.available());
    num = Serial.parseInt();
  }
  return num;
}

void loop() {
  Serial.println("1-127 arası silinecek ID kayıt numarası giriniz: ");
  byte id = numaraOku();
  if (id == 0) { // 0 olmaz.
    return;
  }

  Serial.print("Silinen ID: ");
  Serial.println(id);

  parmakIziSil(id);
}

byte parmakIziSil(byte id) {
  byte p = -1;
```

```
  p = parmakIzi.deleteModel(id);

  if (p == FINGERPRINT_OK) {
    Serial.println("Silindi.");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("İletişim hatası.");
  } else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Bu alandan silinemedi.");
  } else if (p == FINGERPRINT_FLASHERR) {
    Serial.println("Hafızaya yazarken hata.");
  } else {
    Serial.print("Bilinmeyen hata: 0x"); Serial.println(p, HEX);
  }

  return p;
}
```

İşlem Basamakları

1. Görsel 6.13a'daki devreyi kurunuz.
2. Parmak izi kaydetme programını yükleyiniz
3. Seri ekranı açarak yönergeleri takip ediniz.
4. Parmak izi kayıt işlemi bittikten sonra parmak izi okuma programını yükleyiniz.
5. Seri ekranı açınız.
6. Parmağınızı okutarak seri ekranı gözlemleyiniz.

SIRA SİZDE

1. Öğretmeniniz gözetiminde üç ya da dört kişilik gruplara ayrılınız. Görev bölümü yapıp aşağıdaki yönergelere göre devrenin mekanik tasarımını hazırlayınız.

a. Devreyi bir kapıyı açacak şekilde düzenleyiniz.

b. Seri ekran bilgilerini LCD ekrana yazdırınız.

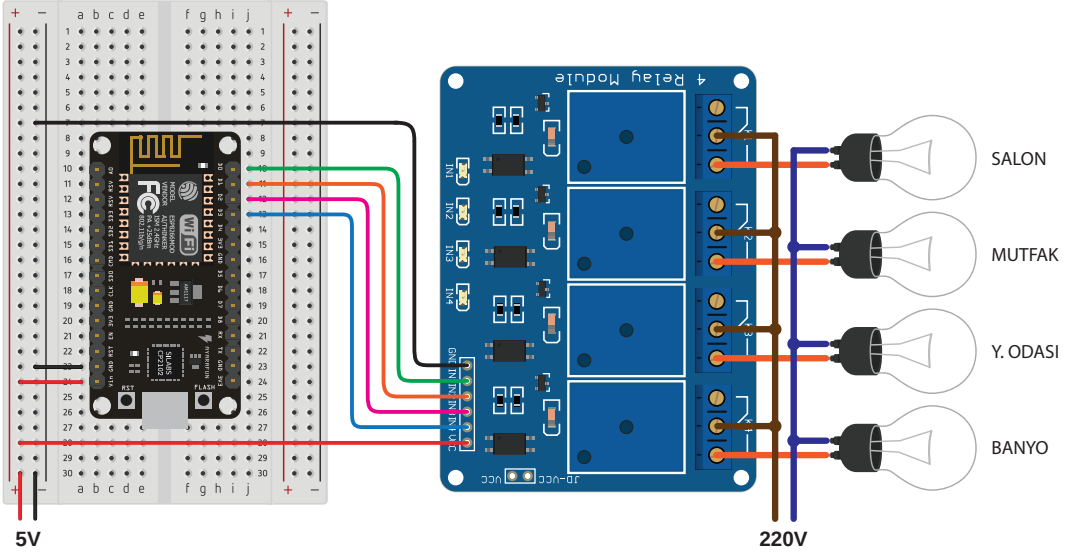
Soru

1. Parmak izi kaydetme, okuma ve yazma programları tek bir programa dönüştürülebilir mi? Deneyiniz.

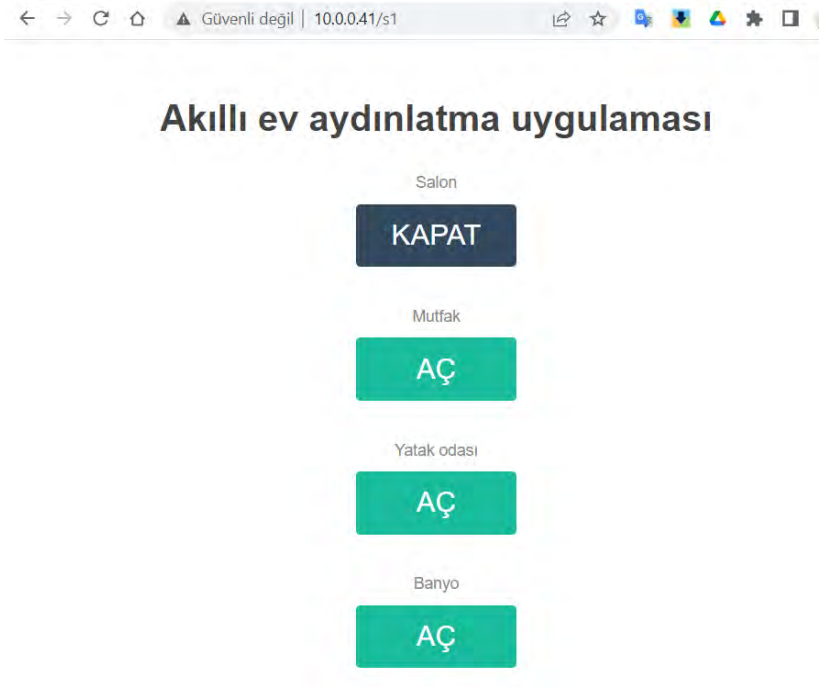
Uygulama Adı: Akıllı Ev Aydınlatma Uygulaması No.: 9

Amaç: Akıllı ev aydınlatma uygulaması yapmak.

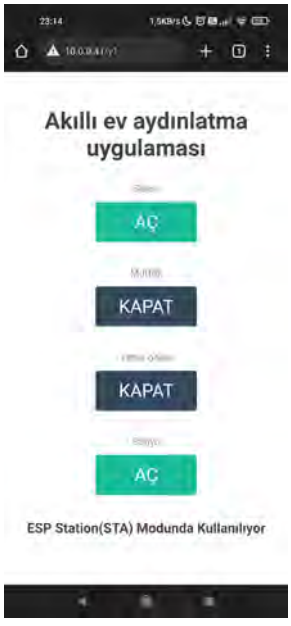
Görsel 6.14'teki devrede NodeMCU ortamdaki kablosuz ağa bağlanmaktadır (STA modu). Seri ekrandan NodeMCU kartının aldığı IP adresi görülür. Bilgisayar (Görsel 6.15) veya akıllı telefondan (Görsel 6.16) tarayıcı adres çubuğuna NodeMCU'nun IP adresi girilerek odaların lamba kontrolleri gerçekleştirilir.



Görsel 6.14: Akıllı ev aydınlatma devresi



Görsel 6.15: NodeMCU'ya bilgisayardan bağlanma



Görsel 6.16: NodeMCU'ya akıllı telefondan bağlanma

Akıllı ev aydınlatma programı aşağıdaki gibidir:

```
#include <ESP8266WiFi.h> // NodeMCU kurulumuyla gelir.
#include <ESP8266WebServer.h> // NodeMCU kurulumuyla gelir.

/*Ağ adınızı ve Şifrenizi Giriniz*/
const char* kablosuzAgAdi = "……";  // Ağ adınızı girin.
const char* sifre = "……";  //Ağ şifrenizi girin.

ESP8266WebServer server(80); /* 80 portunu kullanarak bir server isimli nesne oluşturdu.
*/

const byte salonPin = D0, mutfakPin = D1, yatakPin = D2, banyoPin = D3;
bool salonDurum, mutfakDurum, yatakDurum, banyoDurum;

void setup() {
  Serial.begin(9600);
  delay(100);
  pinMode(salonPin, OUTPUT);
  pinMode(mutfakPin, OUTPUT);
  pinMode(yatakPin, OUTPUT);
  pinMode(banyoPin, OUTPUT);

  Serial.println(kablosuzAgAdi);
  Serial.println(" Ağına Bağlanılıyor");

  WiFi.begin(kablosuzAgAdi, sifre);

  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
Serial.print(".");
  }
  Serial.println("");
  Serial.println("Ağ bağlantısı sağlandı.");
  Serial.print("IP Adresiniz: ");  Serial.println(WiFi.localIP());
  server.on("/", hepsiKapali);
  server.on("/s1", salonAcik);
  server.on("/s0", salonKapali);
  server.on("/m1", mutfakAcik);
  server.on("/m0", mutfakKapali);
  server.on("/y1", yatakAcik);
  server.on("/y0", yatakKapali);
```

```
  server.on("/b1", banyoAcik);
  server.on("/b0", banyoKapali);
  server.onNotFound(sayfaBulunamadi);
 server.begin();
  Serial.println("HTTP Sunucusu Başlatıldı");
}
void loop() {
  server.handleClient();
  if (salonDurum)
    digitalWrite(salonPin, HIGH);
  else
    digitalWrite(salonPin, LOW);

  if (mutfakDurum)
    digitalWrite(mutfakPin, HIGH);
  else
    digitalWrite(mutfakPin, LOW);

  if (yatakDurum)
    digitalWrite(yatakPin, HIGH);
  else
    digitalWrite(yatakPin, LOW);

  if (banyoDurum)
    digitalWrite(banyoPin, HIGH);
  else
    digitalWrite(banyoPin, LOW);
}

void hepsiKapali() {
  salonDurum = LOW;
  mutfakDurum = LOW;
  yatakDurum = LOW;
  banyoDurum = LOW;
  server.send(200, "text/html", HTML(salonDurum, mutfakDurum, yatakDurum, banyoDurum ));
}

void salonAcik() {
  salonDurum = HIGH;
  server.send(200, "text/html", HTML(true, mutfakDurum, yatakDurum, banyoDurum));
}
void salonKapali() {
  salonDurum = LOW;
  server.send(200, "text/html", HTML(false, mutfakDurum, yatakDurum, banyoDurum));
}

void mutfakAcik() {
  mutfakDurum = HIGH;
 server.send(200, "text/html", HTML(salonDurum, true, yatakDurum, banyoDurum));
}

void mutfakKapali() {
  mutfakDurum = LOW;
  server.send(200, "text/html", HTML(salonDurum, false, yatakDurum, banyoDurum));
}

void yatakAcik() {
  yatakDurum = HIGH;
  server.send(200, "text/html", HTML(salonDurum, mutfakDurum, true, banyoDurum));
}

void yatakKapali() {
  yatakDurum = LOW;
  server.send(200, "text/html", HTML(salonDurum, mutfakDurum, false, banyoDurum));
}

void banyoAcik() {
  banyoDurum = HIGH;
  server.send(200, "text/html", HTML(salonDurum, mutfakDurum, yatakDurum, true));
}

void banyoKapali() {
  banyoDurum = LOW;
  server.send(200, "text/html", HTML(salonDurum, mutfakDurum, yatakDurum, false));
}
```

```
void sayfaBulunamadi() {
  server.send(404, "text/plain", "Sayfa Bulunamadı");
}

String HTML(uint8_t salonDurum, uint8_t mutfakDurum, uint8_t yatakDurum, uint8_t banyoDurum)
{
  String metin = "<!DOCTYPE html> <html>\n";
  metin += "<head><meta name=\"viewport\" content=\"width=device-width, initial-
scale=1.0, user-scalable=no\">\n";
  metin += "<meta http-equiv=\"Content-Type\" content=\"text/html;charset=UTF-8\">";
  metin += "<title>Akıllı ev aydınlatma uygulaması</title>\n";
  metin += "<style>html { font-family: Helvetica; display: inline-block; margin: 0px
auto; text-align: center;}\n";
  metin += "body{margin-top: 50px;} h1 {color: #444444;margin: 50px auto 30px;} h3
{color: #444444;margin-bottom: 50px;}\n";
  metin += ".button {display: block;width: 80px;background-color: #1abc9c;border:
none;color: white;padding: 13px 30px;text-decoration: none;font-size: 25px;margin: 0px
auto 35px;cursor: pointer;border-radius: 4px;}\n";
  metin += ".button-on {background-color: #1abc9c;}\n";
  metin += ".button-on:active {background-color: #16a085;}\n";
  metin += ".button-off {background-color: #34495e;}\n";
  metin += ".button-off:active {background-color: #2c3e50;}\n";
  metin += "p {font-size: 14px;color: #888;margin-bottom: 10px;}\n";
  metin += "</style>\n";
  metin += "</head>\n";
  metin += "<body>\n";
  metin += "<h1>Akıllı ev aydınlatma uygulaması</h1>\n";

  if (salonDurum)
    metin += "<p>Salon</p><a class=\"button button-off\" href=\"/s0\">KAPAT</a>\n";
  else
    metin += "<p>Salon</p><a class=\"button button-on\" href=\"/s1\">AÇ</a>\n";

  if (mutfakDurum)
    metin += "<p>Mutfak</p><a class=\"button button-off\" href=\"/m0\">KAPAT</a>\n";
  else
 metin += "<p>Mutfak</p><a class=\"button button-on\" href=\"/m1\">AÇ</a>\n";
  if (yatakDurum)
    metin += "<p>Yatak odası</p><a class=\"button button-off\" href=\"/y0\">KAPAT</a>\n";
  else
    metin += "<p>Yatak odası</p><a class=\"button button-on\" href=\"/y1\">AÇ</a>\n";
  if (banyoDurum)
    metin += "<p>Banyo</p><a class=\"button button-off\" href=\"/b0\">KAPAT</a>\n";
  else
    metin += "<p>Banyo</p><a class=\"button button-on\" href=\"/b1\">AÇ</a>\n";

  metin += "</body>\n";
  metin += "</html>\n";
  metin += "<h3>ESP Station(STA) Modunda Kullanılıyor</h3>\n";
  return metin;
}
```

İşlem Basamakları

1. Görsel 6.14'teki devreyi kurunuz.
2. Programı yükleyerek seri ekrandan IP adresini okuyunuz.
3. Bilgisayardan veya akıllı telefonun tarayıcısına IP adresini yazıp karta bağlantı sağlayınız.
4. Sayfa arayüzündeki butonlarla lambaları kontrol ediniz.

SIRA SİZDE

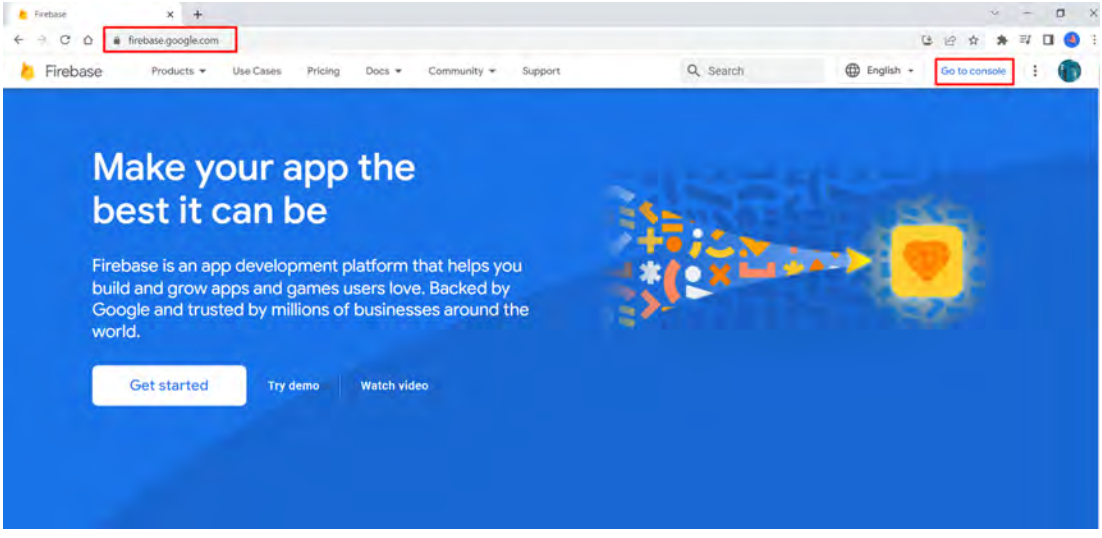
1. Devreye ve programa evdeki diğer odaların lambalarının kontrolünü de ekleyiniz.

Soru

1. Lambalar internet üzerinden kontrol edilebilir mi? Araştırınız.

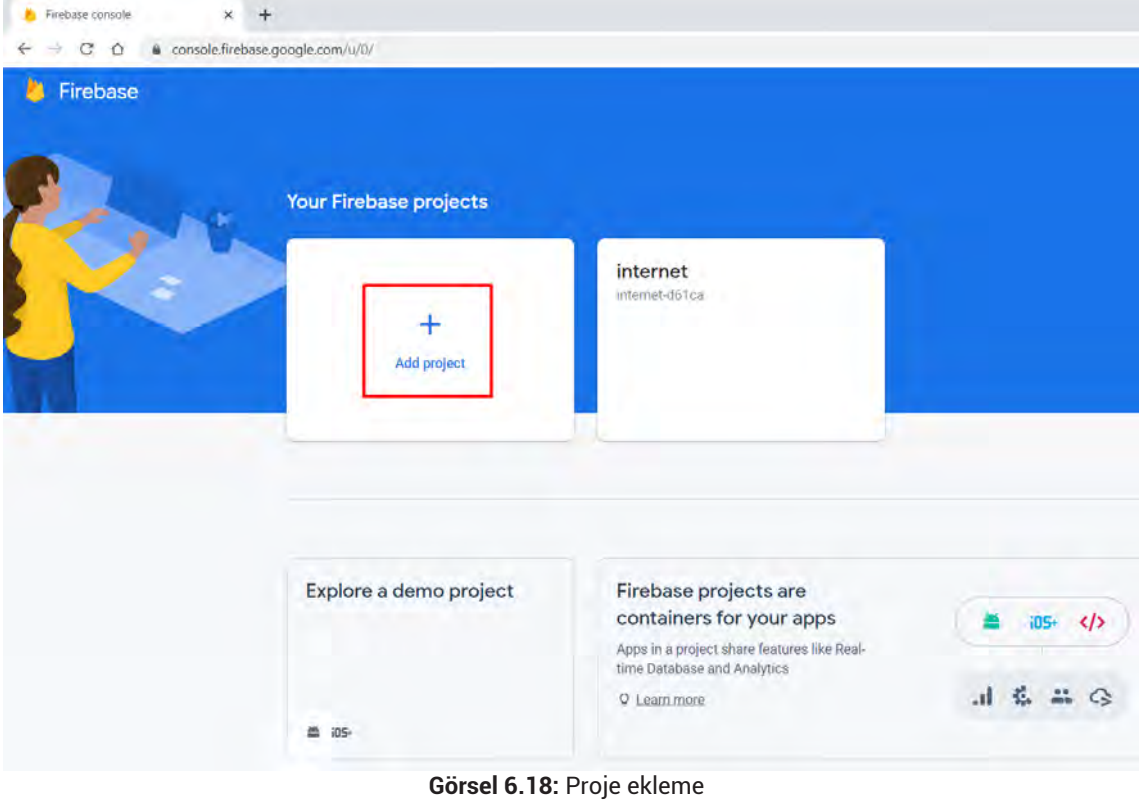
Google Firebase Kullanımı

NodeMCU intenette herhangi bir server üzerinde database kullanılarak kontrol edilebilir. Google firebase ücretsiz bir database uygulamasıdır ve ESP kartları için yazılmış kütüphanelere Arduino IDE üzerinden ulaşılabilir. Database oluşturmak için **firebase.google.com** adresine Google hesabıyla giriş yapılarak "Got to console" tıklanır (Görsel 6.17).



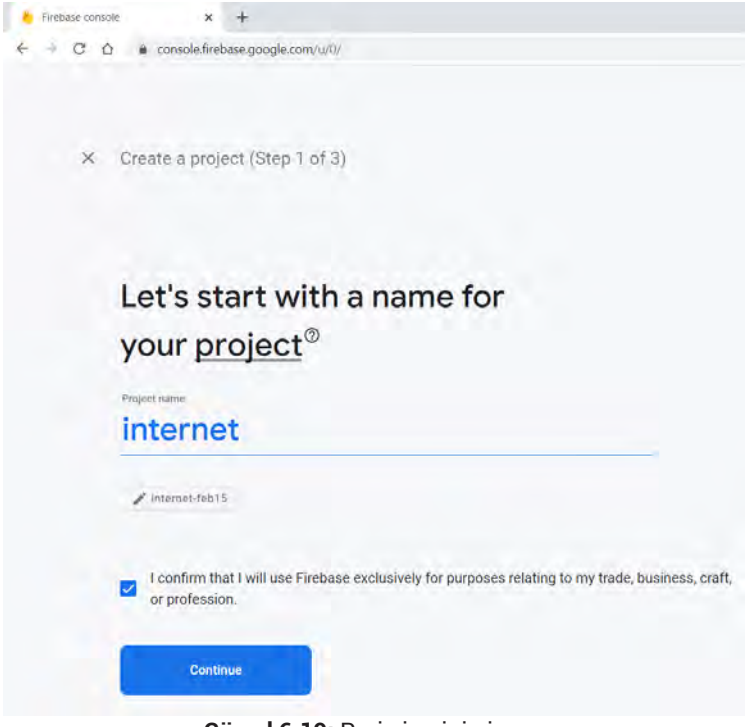
Görsel 6.17: Firebase konsoluna giriş

"Add Project" tıklanır (Görsel 6.18).



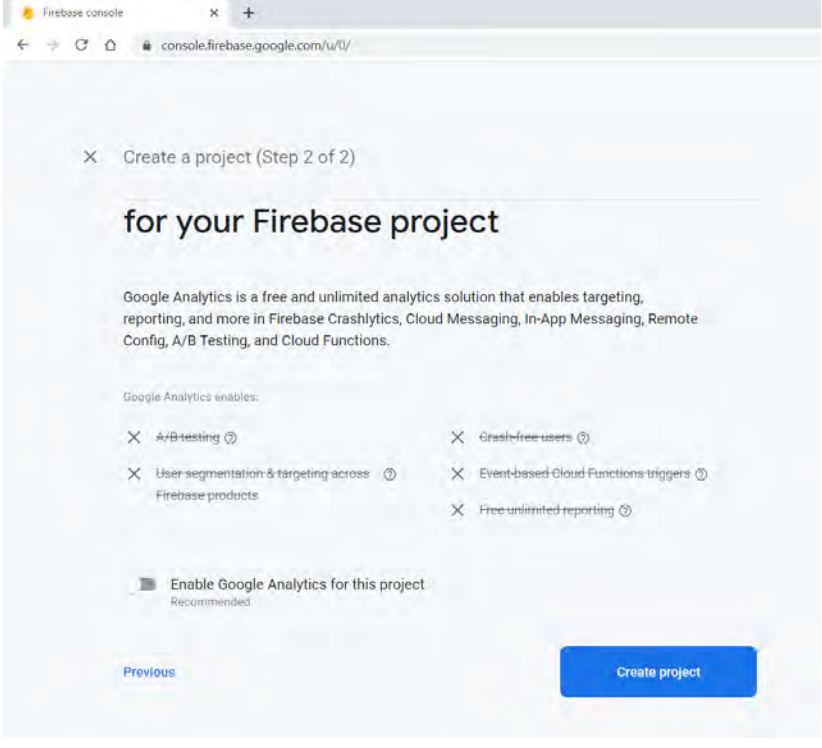
Görsel 6.18: Proje ekleme

Proje ismi girilerek "Continue" düğmesi tıklanır (Görsel 6.19).



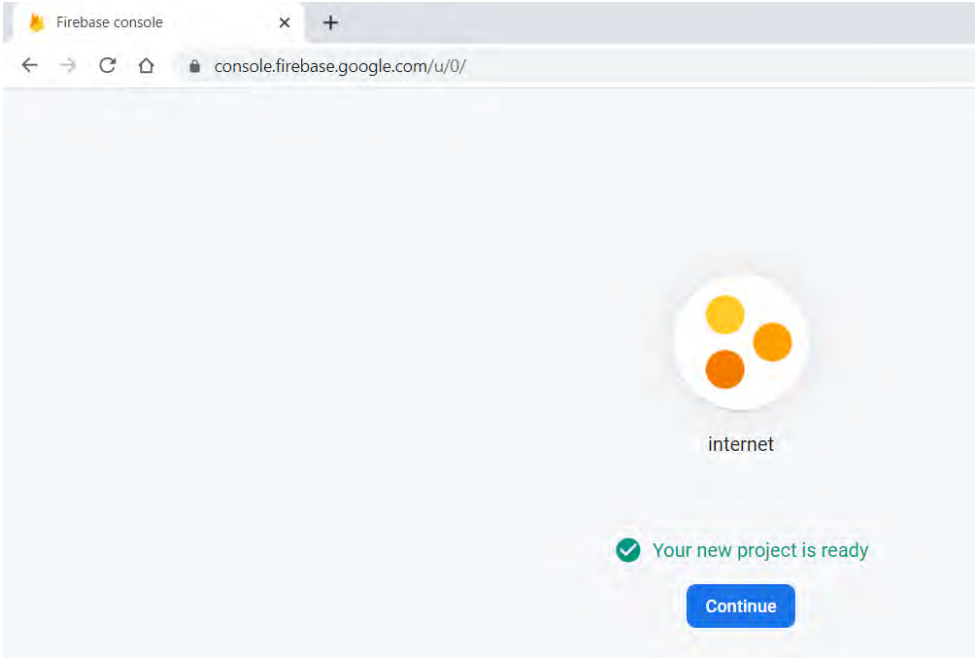
Görsel 6.19: Proje ismini girme

Sonraki sayfada "Create Project" tıklanarak database oluşturulur (Görsel 6.20).



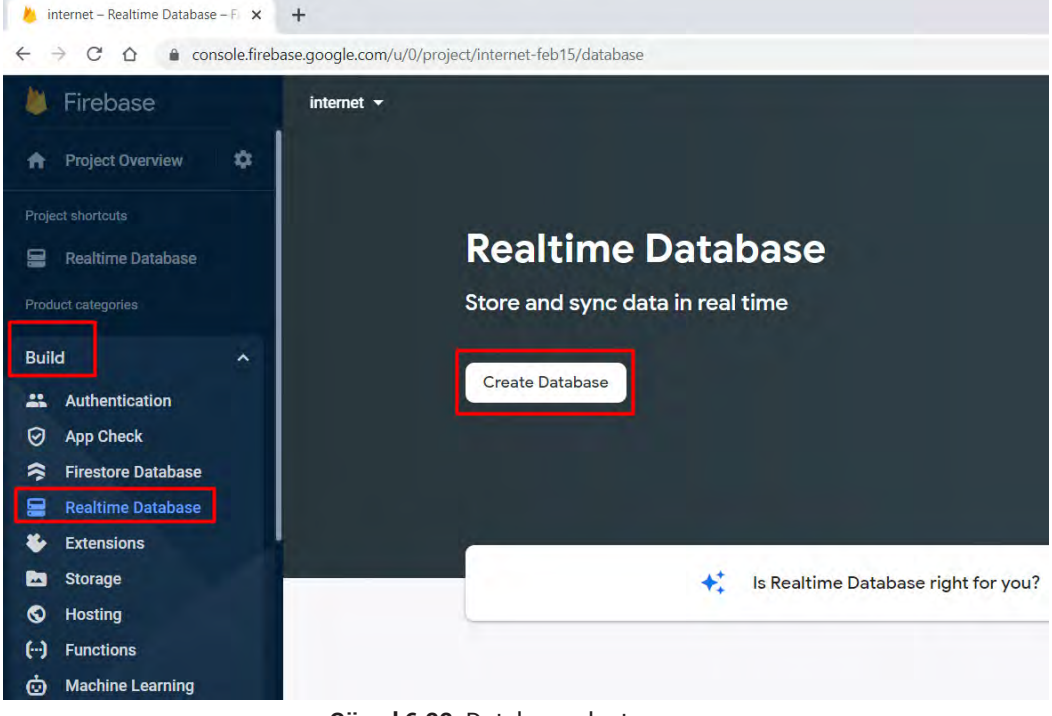
Görsel 6.20: Database oluşturma

Görsel 6.21'de görülen "internet" isimli proje oluşturuldu. Proje oluşturulduktan sonra devamında "Continue" düğmesi tıklanır.



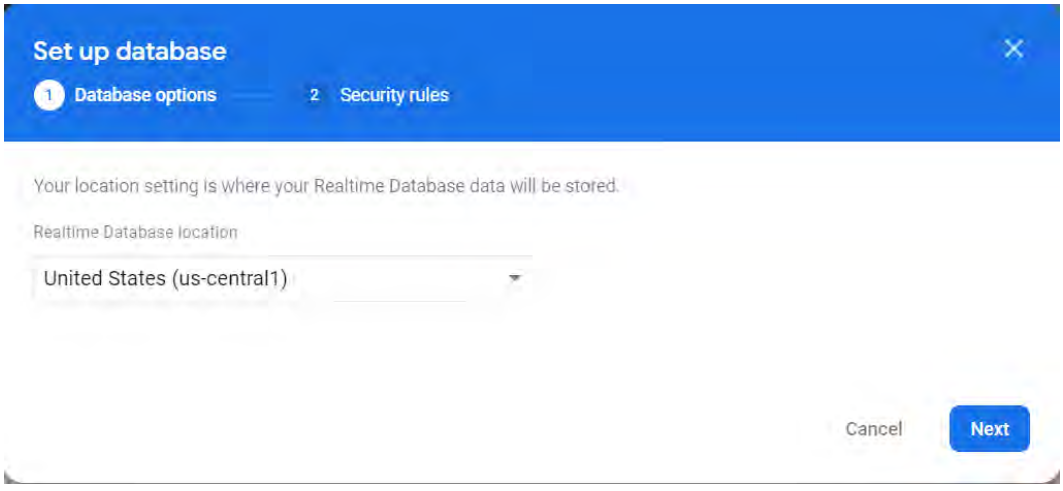
Görsel 6.21: Proje oluşturma

Proje sayfasında "Build Realtime Databse → Create Database" adımları izlenir (Görsel 6.22).



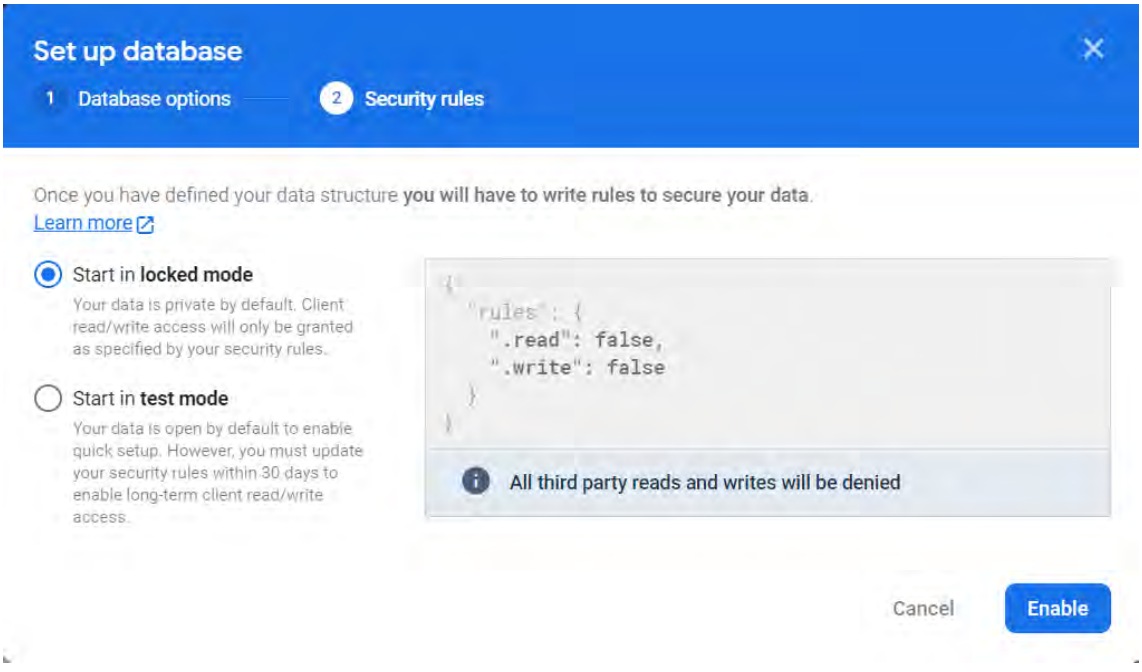
Görsel 6.22: Database oluşturma

Database lokasyonu seçilir ve “Next” düğmesi tıklanır (Görsel 6.23).



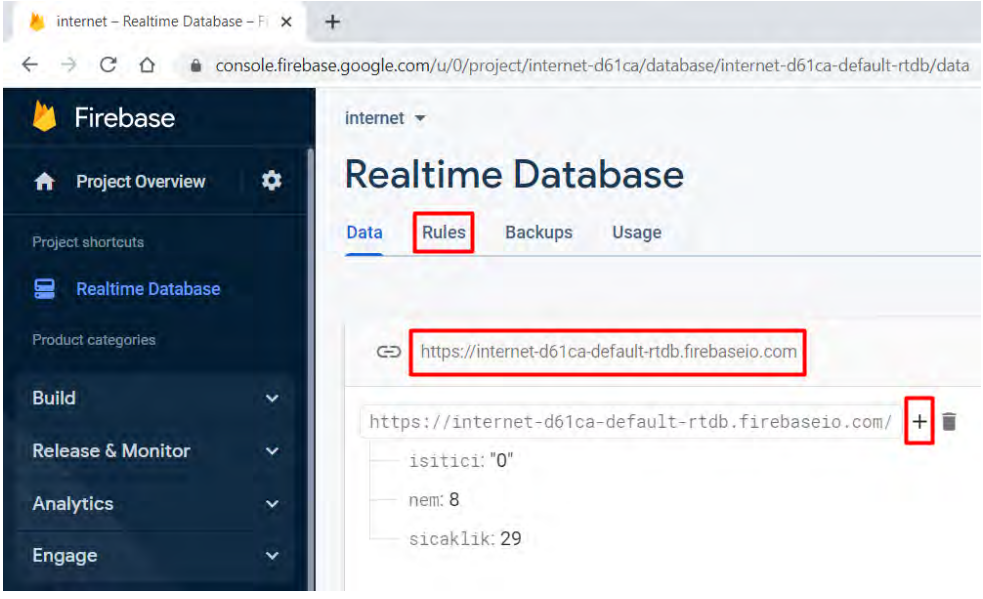
Görsel 6.23: Lokasyon seçme

Sonraki sayfada “Enable” düğmesi tıklanır (Görsel 6.24).



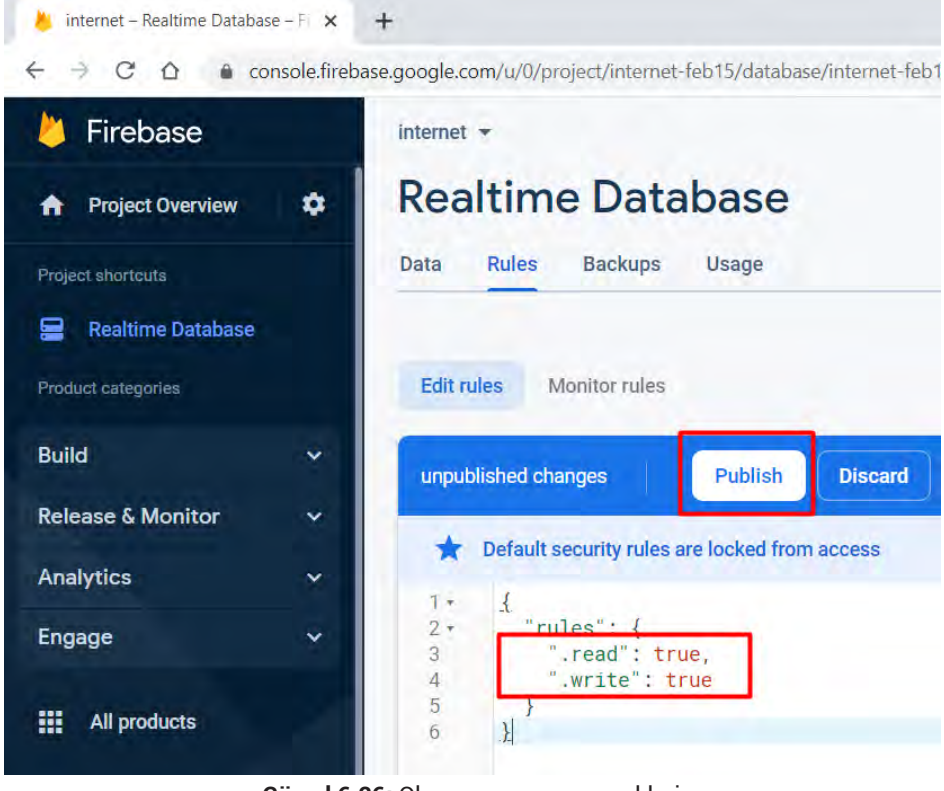
Görsel 6.24: Güvenlik seçenekleri

Realtime Database sayfasında database adresinin (Bu adres programda ve akıllı telefon uygulamasında kullanılacaktır.) yanındaki "+" simgesiyle değişkenler tanımlanır. Daha sonra "Rules" düğmesi tıklanır (Görsel 6.25).



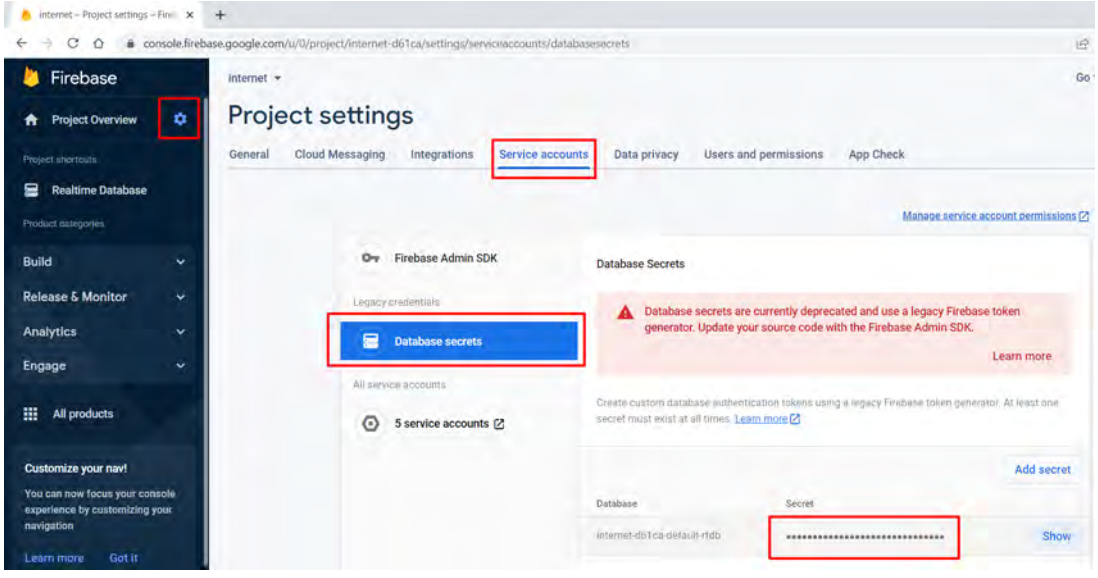
Görsel 6.25: Değişken tanımlama

Okuma ve yazma seçenekleri "true" yapılarak "Publish" düğmesi tıklanır (Görsel 6.26).



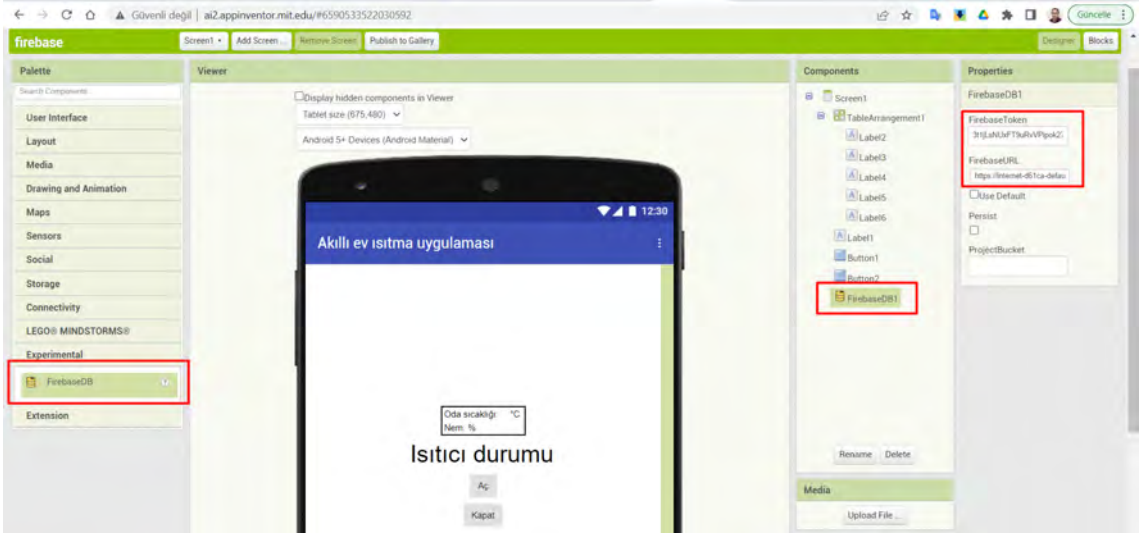
Görsel 6.26: Okuma yazma seçenekleri

"Project settings →Service account →Database secrets" seçenekleri izlenerek "Show" düğmesi tıklanır. Bu gizli anahtar kopyalanarak programda ve akıllı telefon uygulamasında ilgili yerlere yapıştırılır (Görsel 6.27).



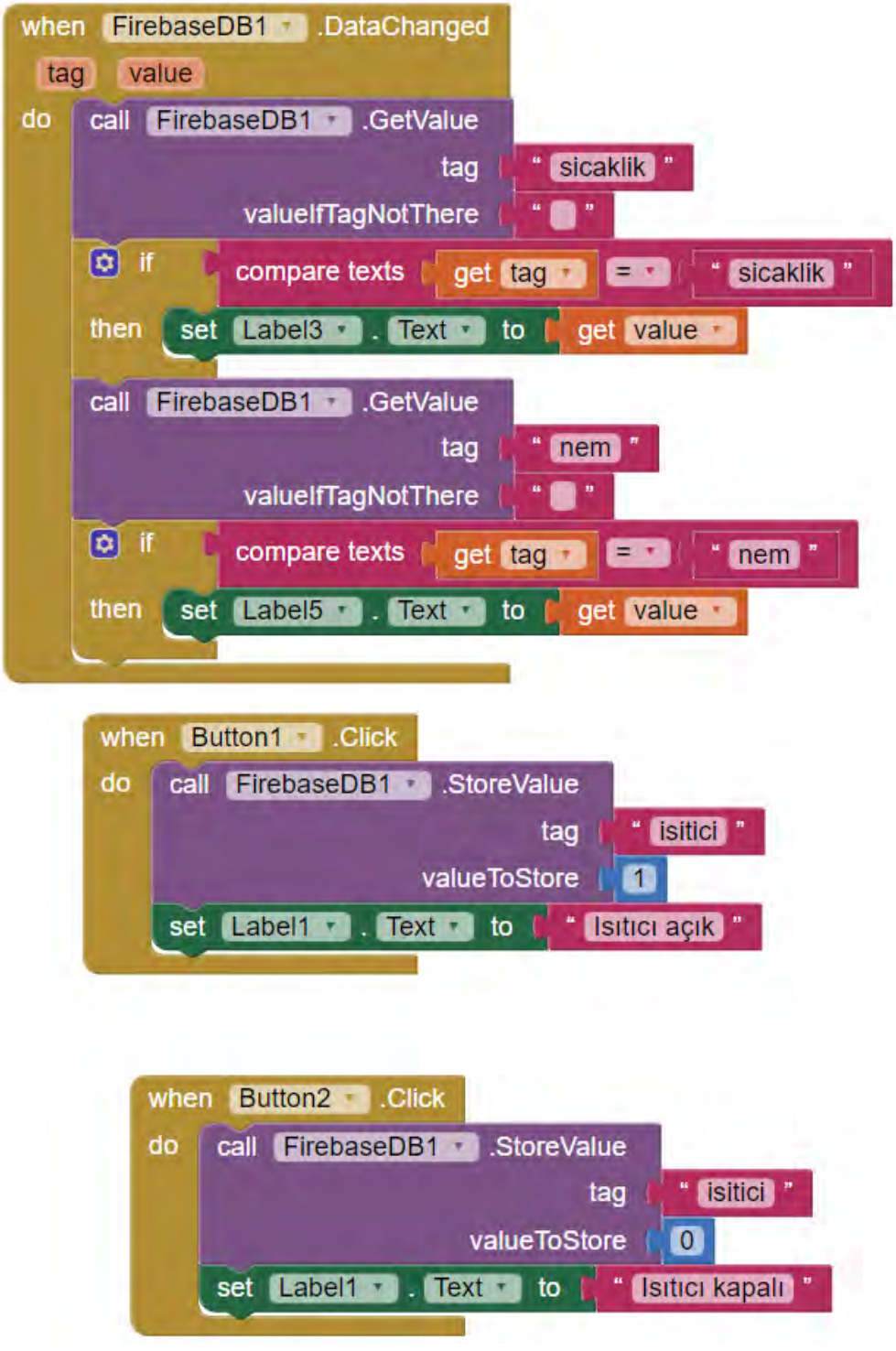
Görsel 6.27: Gizli anahtar

Mit app invertorde Experimantal kısmından FirebaseDB alınır. Firebase kompanenti tıklanarak firebase adresi ve gizli anahtarı girilir (Görsel 6.28).



Görsel 6.28: Akıllı telefon uygulaması oluşturma

Görsel 6.29'daki kodlarla akıllı telefon uygulaması oluşturulur.

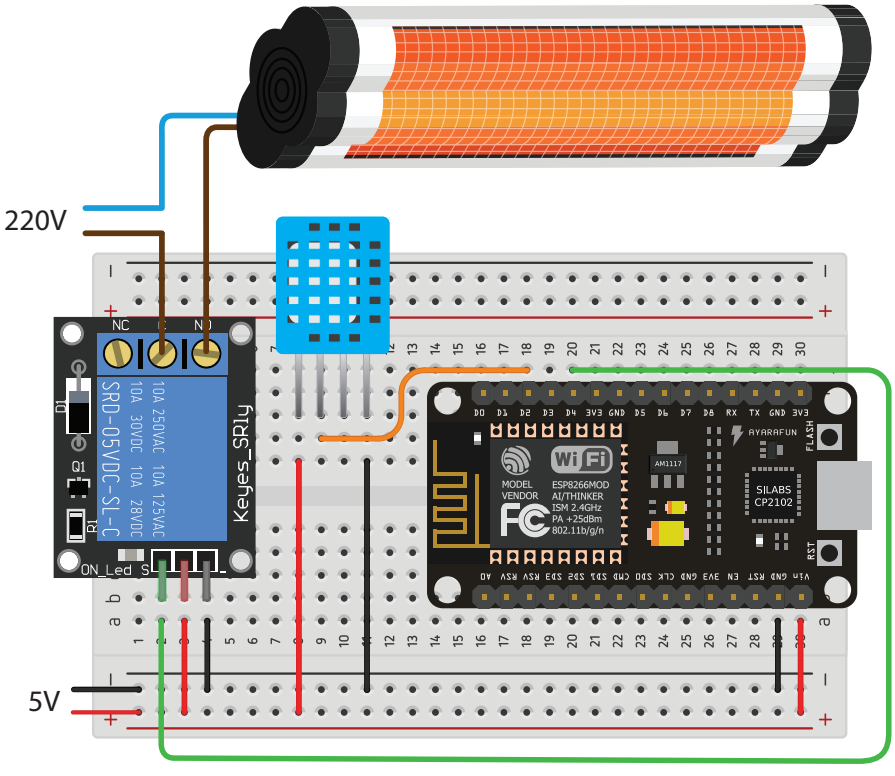


Görsel 6.29: Mit app invertor kodları

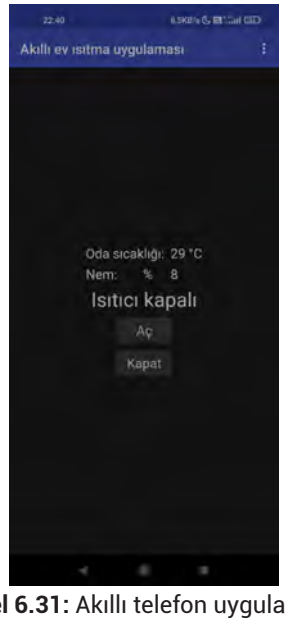
Uygulama Adı: Akıllı Ev Isıtma Uygulaması — No.: 10

Amaç: Akıllı ev ısıtma uygulaması yapmak.

Görsel 6.30'daki devre, bulunduğu ortamın nem ve sıcaklık bilgisini firebase veri tabanına yazar. Görsel 6.31'deki akıllı telefon uygulamasından bu değerler okunur. NodeMCU kartına bağlı ısıtıcı internet üzerinden kontrol edilir.



Görsel 6.30: Akıllı ev ısıtma devresi



Görsel 6.31: Akıllı telefon uygulaması

Akıllı ev ısıtma programı aşağıdaki gibidir:

```
#include <ESP8266WiFi.h> // NodeMCU kurulumuyla gelir.
#include <FirebaseESP8266.h> // FirebaseESP8266 Client v3.8.3 ve Firabasejson v2.6.3
#include "DHT.h" // Adafruit DHT Sensor Library v1.2.3

#define DHTpin 4     // Sensörün bağlandığı pin. NodeMCU 1.0 (ESP-12E) D2 pini.
#define DHTtipi DHT11   // Sensör tipi: DHT11, DHT21, DHT22.

DHT dht(DHTpin, DHTtipi); //dht isimli nesne oluşturuldu.

#define WIFI_SSID "....."  // Kablosuz ağ adı.
#define WIFI_PASSWORD "....."  // Kablosuz ağ şifresi.

#define DATABASE_URL "internet-d61ca-default-rtdb.firebaseio.com" // Firebase'de oluşturu-
lan nesnenin adresi.

FirebaseData veritabani; // Veri tabanı isimli nesne oluşturuldu.

FirebaseAuth auth;
FirebaseConfig config;

void setup() {
  pinMode(2, OUTPUT);      //  GPIO2 pini D4 pini çıkış.
  pinMode(4, INPUT);      //  GPIO4 pini D2 pini giriş.
  dht.begin();

  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

  config.database_url = DATABASE_URL;
  config.signer.tokens.legacy_token = "3t1jLsNUxFT9uRvVPipok27rF7b25RtBQIcbn9Pn"; //
Gizli anahtar (token).

  Firebase.begin(&config, &auth);
}

void loop() {
  int nem = dht.readHumidity(); // Nem bilgisini oku.
  int sicaklik = dht.readTemperature(); // Sıcaklık bilgisini oku.

  Firebase.setInt(veritabani, "/nem", nem); // Nem bilgisini firebase veri tabanına yaz.
  Firebase.setInt(veritabani, "/sicaklik", sicaklik); // Sıcaklık bilgisini firebase veri
tabanına yaz.

  if (Firebase.getString(veritabani, "/isitici")) { // Isıtıcın durum bilgisini firebase
veri tabanından al.
    String metin = veritabani.to<String>();
    Serial.println(metin);
    if (metin == "1")
      digitalWrite(2, LOW);
    else
      digitalWrite(2, HIGH);
  }
}
```

İşlem Basamakları

1. Görsel 6.30'daki devreyi kurunuz.
2. Akıllı telefon uygulamasından devrenin bulunduğu ortamın nem ve sıcaklık bilgisini gözlemleyiniz.
3. Isıtıcıyı internetten kontrol ederek sıcaklığı gözlemleyiniz.

SIRA SİZDE

1. Devrenin klimayla bağlantısını sağlayarak soğutma yapınız.

Sorular

1. Web sayfası üzerinden akıllı ev kontrolü nasıl yapılır? Araştırınız.

NOT DEFTERİ

KAYNAKÇA

Adafruit Industries. (2022). *Adafruit Optical Fingerprint Sensor*. New York.

InvenSense Inc. (2013). *MPU-6000 and MPU-6050 Product Specification*. U.S.A.

Küçükkoç, İ. (2020). *Algoritma ve Programlamaya Giriş*. Balıkesir.

Microchip Technology Inc. (2018). *ATmega48A/PA/88A/PA/168A/PA/328/P.*

Nordic Semiconductor. (2008). *nRF24L01+ Single Chip 2.4GHz Transceiver Preliminary Product Specification.* Trondheim.

Pololu Corporation. (2014). *QTR-8A and QTR-8RC Reflectance Sensor Array User's Guide*.

T.C. Millî Eğitim Bakanlığı. (2012). *Mikroişlemci ve Mikrodenetleyiciler*. Ankara.

GENEL AĞ KAYNAKÇASI

- https://www.arduino.cc/ (27.07.2021 saat 16.20)
- https://lastminuteengineers.com/ (24.04.2022 saat 22.14)
- https://dronebotworkshop.com/ (12.05.2022 saat 20.48)
- https://randomnerdtutorials.com/ (14.06.2022 saat 20.10)
- https://app.code2flow.com/ (21.06.2022 saat 10.35)
- https://learn.adafruit.com/ (15.06.2022 saat 18.12)
- https://maker.robotistan.com/ (04.06.2022 saat 10.32)
- http://www.temrinler.com/ (12.08.2021 saat 12.10)