Bu yazımda, IR uzaktan kumanda kodlarını kolay bir şekilde nasıl bulabileceğinizi anlattım. Arduino kullananlar bilirler. IR kütüphanesi vardır ve o programı kullandıklarında, kumanda tuşunun kodunu seri monitörde görürler. Zaten kolayca bunu yapabiliyorken bu yazının anlamı ne diyebilirsiniz.   
Anlattığım yöntem ile kütüphaneye ihtiyacınız olmadığı gibi kütüphanelerde bulunan hatalardan dolayı yanlış tuş kodu geldiğinde, bu kodu doğru zannedip gelen kodu o kumandanın gerçekten kontrol ettiği cihazda denenmesi durumunda hata vereceğidir. Ayrıca herkes kendi işlemcisinde IR kütüphanesine sahip olamayacaktır. Fakat anlattığım yol ile kendi işlemcisine kolaylıkla uyarlayabilecektir.

Kütüphanelerin tamamı böyle hatalı diyemem Fakat elimdeki kütüphane seri monitörde gösterirken veya yakalarken hata yapıyor. Bu hata sizin arduino projelerinizi etkilemiyor. Çünkü algoritma, devamlı aynı yerde aynı hatayı yaptığından bulduğu kod hep aynı oluyor. Bu şekilde ise siz gelen kodu kullanmak istediğinizde sorun yaratmadan tekrar, tekrar aynı kodu bulacağından sizin projenizi etkilemeyecek.   
Bunun yanı sıra aldığınız kodu protokolüne uygun kumandadan gelmiş gibi simüle edip gerçek cihazı çalıştırmak isteseniz, Gelen kod yanlış olduğundan dolayı kontrol edemeyeceksiniz.

**Kütüphanesiz çalışmada başlıca 2 yöntem izlenebilir.**   
Kolay olanı ki benim kullandığım bu. PULSIN komutu. Bu komut ile isterseniz HIGH seviyesinin ne kadar süre devam ettiğini, isterseniz LOW seviyesinin ne kadar süre devam ettiğini, mikro saniye olarak öğrenebilirsiniz.   
Zor olanı ise timer kesmesini kullanarak protokole uygun zaman aralıklarında sinyalin durumuna bakmak.   
Her iki yoldan farklı çözümler de bulunabilir. Fakat ben programlarımda PULSEIN komutunu tercih ettim.

**PULSIN örnek program koy**

Protokollere geçmeden önce sinyalin nasıl okunması gerektiğini anlatmak istiyorum. İnternette NEC protokol dediğinizde size herhangi bir tuşa ait sinyal resmi gelecektir. Fakat gelen sinyal resmi Arduinoya gelen sinyalin tersi olarak görünecektir. Sebep ise arduino alıcı gözün içindeki devre gelen sinyali terslemektedir. Kütüphaneler bu terslenmiş sinyale göre kodu çözer. Kendi programınızı bu terslenmiş sinyale göre yapabilirsiniz ( Ben terslenmiş sinyale göre yaptım ) Ya da sinyali tek bir transistör kullanarak tekrar tersleyerek orijinal sinyale dönüştürüp programınızı ona göre yapabilirsiniz.

**Buraya tersleme sinyali ve transistörlü devreyi koyup açıkla.**

Doğrusu hangisi derseniz orijinal sinyalin şekline uygun olandır. Kendi örnek programlarımda kolaylık olsun diye terslenmiş sinyali kullandım.

Ayrıca sinyal resimlerine bakarken LSB ve MSB göreceksiniz. Bu binary sayı sistemlerinde en yüksek bit ve en düşük biti tanımlamaktadır.   
Mesela BINARY 10001001 sayısını ele alırsak en sağdaki bit LSB dir. En soldaki bit ise MSB dir.   
Sinyal grafiği protokole göre verildiğinden bazı protokollerde en düşük bit önceden gönderilir bazılarında ise en düşük bit en son gönderilir.

**LSBMSB resmini ekle**

Bu neden önemli derseniz, gelen 1 ve 0 lardan oluşmuş binary sayının 1 ler basamağı neredeyse ona göre sayıyı hesaplamanız için. Ayrıca matematikte soldaki sıfırın bir önemi yoktur. Bazı binary yazan programlar ( mesela kullandığım kütüphane gibi) soldaki 0 değerini atıp gösteriyor. Bu sıfır MSB deyse sorun olmayacaktır. Fakat LSB de ise binary sayı hatalı çıkacaktır çünkü 1 ler basamağı yoktur.

Bununla ilgili bir örnek kod aşağıdadır.   
01000000101111111000000001111111 burada en soldaki sıfırı kütüphane göstermiyor. Fakat biz LSB kısmı en solda diye soldan itibaren kodları ayırmaya başlarsak, soldan ilk 8 bit diyoruz ve 0 orada görünmüyorsa ilk bit 1 den başlar dediğimizde ise hata yapmış olacağız. Benim yaptığım program da ise 32 bitin tamamı sıfır bile olsa oraya yazılır. Bu da size ayırmada kolaylık sağlar. Bununla ilgili örnekler aşağıda verilecektir.

**Not:** **Arduino alıcı göz modülü (KY-022)** 38khz taşıyıcı frekansına göre filtrelenmiş gibi görünse de 36Khz sinyalleri de alabiliyor.   
**Bununla birlikte Alıcı gözün sinyali terslediğini de unutmamamız kazım.**

**KY-022 modül resmi. Ve arduino ile bağlantısı**

**Taşıyıcı frekans ne demek:** IR sinyallerin daha uzak mesafelere gidebilmesi ve ayrıca IR verici led akımının daha yüksek akımda anahtarlamalı çalışabilmesi için taşıyıcı frekans kullanılır. DC olarak 20-30 ma akıma dayanan bir LED 38Khz de anahtarlandığında 100ma civarında akım da kullanılabilmektedir. Bu da verici sinyalini kuvvetlendirecek ve mesafeyi arttıracaktır. Bazı alıcı gözler 36Khz frekansına maksimum geçirgenlik için ayarlanmıştır. Bazıları 38Khz bazıları 40 Khz sinyale göre dir. Belki daha farklı modeller de vardır. En çok kullanılan 36Khz ve 38Khz dir.   
Buradaki alıcı göz yakın mesafe denendiği için tüm kumandalarda olumlu sonuç verdi. Fakat 36Khz bir kumanda ile bu 38Khz lik alıcı göze sinyal gönderdiğinizde yakın mesafede sorun olmasa bile mesafe uzadıkça sorun olmaya başlayacaktır. Halbuki 36Khz lik bir kumandayı 36Khz lik bir alıcı gözle kullansanız daha uzak mesafeden sinyal gönderebilirsiniz.   
Taşıyıcı frekans sinyali nasıldır. Alıcı göze resimdeki üst tarafta bulunan sinyal gelir.

**Taşıyıcı frekans resmini koy**

Taşıyıcı frekans arka arkaya geldiğinde HIGH seviyesi hiç gelmediğinde LOW seviyesi olur. Alıcı göz taşıyıcı frekansı yok ederek gerçek sinyali bizlere verir. Bu sinyali direkt olarak gönderici Led’e verip göndereyim deseniz mesafe çok küçük olacak. Fakat taşıyıcı frekansa bindirip gönderdiğinizde daha uzun olacaktır. Bundan dolayı Kumanda vericisi yapılırken kontrol sinyali ve taşıyıcı frekans birleştirilerek gönderilir. Alıcı gözlerde uygun taşıyıcı frekansı süzerek size kontrol sinyalini verir ve bu kontrol sinyalini alıcı devrede işlersiniz.   
Bu resimdeki devre ise basit olarak taşıyıcı frekans ile kontrol sinyali nasıl birleştirildiği(modüle edildiği) görülmektedir.

**Basit transmitter devresi.**

Devre çalışmasına bakarsak diyot geçirgenliği taşıyıcı frekansın LOW kısmında transistor ün beyz ucunu low durumuna getirir. HIGH seviyesinde ise diyot tıkama yönünde çalışacağından kontrol sinyalinin durumuna bağlı olarak transistorün çıkış vermesini sağlar. Mesela kontrol sinyali HIGH olduğunda çıkışta taşıyıcı frekansın da görülmesi gerekir. Bu durum diyota gelen taşıyıcı frekans sinyalinin HIGH ve LOW durumuna göre çıkışa yansıtılır. Bu durum tablodan daha kolay anlaşılmaktadır. Tabloda sarı ile gösterilen alan taşıyıcı frekansın karşıya iletildiği alandır. Diğer alanda ise çıkış LOW durumundadır. Çünkü Kontrol sinyali LOW durumundadır.

**TRansmitter modul tablosu**

NEC PROTOKOLÜ :

**NEC protokol sinyali ve osilaskopta çıkan görüntüyü koy**

NEC protokolü birçok kumandada kullanılıyor gibi. Hatta Samsung kumanda, NEC protokolü kullanmasına rağmen baştaki 9ms start palsını 4.5ms ye olarak kullanmış. Başka NEC benzeri kumandalar da mevcut. Üreticiler temelde NEC protokolü kullansalar da, belkide kumandalar karışmasın diye kendince küçük ayrıntılar koymuşlar. Bunlar NEC protokol temelli olduklarından programlama NEC protokolüne göre yapılacaktır. Sadece aradaki süreler değişecektir. NEC protokolü 32 bitliktir. Start palsi hariç. 32 bit soldan sağa doğru gidilirse   
Adress+adresin tersi+cmd+cmd tersi şeklinde 4 byte lık bir bilgidir. Ayrıca tuş basıldığı sürece tuşun tekrar ettiğini gösteren sadece start palsinden oluşan sinyaller gelmektedir. 1. Sinyal katarı adres ve komuttan oluşan 32 bit bilgidir. Arkasındakiler ise tuş bırakılasıya kadar gelen ve start palsinden oluşan tekrar (repeat) sinyalleridir.

**NEC protokolünün sinyal şekilleri ve anlatımı.**

NEC protokolünde HIGH ve LOW şekilleri birbirinden kolayca ayrılabilir. Şayet 1 periyot içinde HIGH ve LOW süreleri eşitse 0 olarak değerlendirilir. Şayet HIGH ve LOW süreleri farklıysa 1 olarak değerlendirilir. Resimdeki sinyal şeklinde 1 ve 0 durumlarını hemen fark edebileceksiniz.   
Bu süreler kontrol edilirken alıcı gözün terslediğini de dikkate alarak sinyalin hangi kısmına göre karar vermeniz gerektiğini bulabilirsiniz. Orijinal sinyale göre karar vereceksek, LOW seviyesi büyük olana 1 dememiz gerekir. Şayet Alıcı gözün terslemesine göre karar vereceksek, HIGH seviyesi büyük olana 1 dememiz gerekir.   
Bu durum aşağıdaki sinyal şekillerinde görülmektedir. İlk sinyal Terslenmemiş (noninvert) durumu gösterir. 2. Sinyal ise gerçek devrede alıcı gözde terslenmiş (invert) sinyaldir. ( Bu tüm protokoller için aynıdır. Tüm protokollere ait programlar alıcı gözün terslediği şekle göre yapılmıştır)

SAMSUNG NEC PROTOKOLÜ :

Samsung noninvert, invert sinyaller.  
Samsung ta NEC protokolünü kullanıyor. Fakat bazı farklılıklar oluşturmuş durumda   
NEC protokolü : 9ms+4.5ms+adres+adres tersi+komut+komut tersi  
Samsung protokol : 4.5ms+4.5ms+adres+adres+komut+komut tersi   
Şeklindedir. Fakat NEC protokolünde kullandığınız programın aynısını kullanabilirsiniz. Sadece ilk start palsindeki 9ms yerine 4.5ms ile başladığından start palsı yakalama komutunun süresini 4.5ms yapmanız yeterli. Kodlama kısmında analiz ederken zaten adresin tersi ve komutun tersini dikkate almayacağımız için yine aynı şekilde hesaplanabilir.   
  
NEC ve SAMSUNG noninvert şekillerini buraya koy.

PHILIPS RC5 protokolü:  
Bu protokolü yakalamak kolay fakat hesaplamak biraz daha zor görünüyor. Çünkü gelen sinyalde start palsı, HIGH seviyesi ve LOW seviyesi hep aynı sürededir. Burada kullanılan kodlama **Manchester kodlama sistemidir**. Bu kodlama sisteminde LOW pozisyondan HIGH pozisyona çıkıyorsa 1, HIGH seviyesinden LOW seviyesine düşüyorsa 0 seviyesi olarak değerlendirilir.

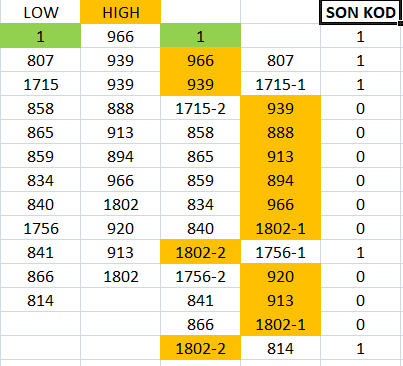
**Kodlama 1 ve 0 seviye resmini koy. Manchester kod resmi ve orijinal sinyali bir de noninvert sinyalini koy**

Yukarıdaki ilk resimde Manchester kodlamanın 2 tip standardı olduğunu görüyorsunuz. Bizler RC5 orijinal sinyaline göre işlem yapıyorsak IEEE 802.3 standart yapısına göre kodlamayı yapacağız. Fakat sizinde farkına varacağınız gibi alıcı gözün terslediği sinyale göre bakarsak LOW seviyesinden HIGH seviyesine çıkarken 0 diyoruz. Bu da G. E. Thomas standardına göre kullandığımızı göreceksiniz. Yani alıcı göz terslediğinde standart değişmiş oluyor. Buna dikkat etmeniz gerekiyor. Ayrıca RC5 protokolünde start palsından sonra verilerin MSB yönünden gönderilmeye başlanır. Buna dikkat edin. NEC protokolünde ise start palsinden sonra LSB yönünde bilgiler gönderilir.

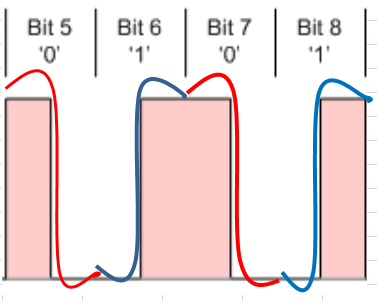
Bunun yanı sıra diğer protokollerden farklı olarak bu protokol 36Khz taşıyıcı frekans kullanıyor. Yukarıda da anlattığım gibi Alıcıda 36Khz lik göz kullanıldığında maksimum mesafeye ulaşılabilir. Arduino IR alıcı modülü 38Khz e göre ayarlıdır. Fakat 36Khz de de sinyalleri alabilmektedir. Yakın mesafe olduğundan.

Ayrıca bu protokolde bir de toggle biti vardır. Her tuşa bastığınızda HIGH ise LOW, LOW ise HIGH durumuna geçerek tuşa ilk defa bastığı belli olur. NEC protokolünde tekrar sinyali vardı. Burada ise yine tuş basılı olduğu anda kendini tekrar eder ama sadece start palsı değil aynı sinyali tekrar eder ve toggle bitini değiştirmez. Şayet butonu bırakıp aynı tuşa bile bassanız toggle biti değişecektir.   
**RC5 sinyalinin resmini koy ve toggle bitini işaretle.**

Programını yaparken yine aynı şekilde PULSIN komutu kullanılır. HIGH ve LOW seviyelerinin süreleri alınır. Bu süreler alındığında durum aşağıdaki gibidir.



Yukarıdaki tabloda gelen sinyalin HIGH ve LOW seviyelerinin süreleri en soldaki sütunlardır. En baştaki yeşil 1 rakamı start palsının yakalanması sırasında süresi hesaplanmayan 1 dir ve kodlama da en başa yazılmıştır.   
Daha sonra ise sinyal şekline bakarsak hangi sinyal geldiyse o sinyali 3. Ve 4. Sütunlara yazdık. Bundan sonra ise **Manchester** kodlamaya göre 1 önce yazılmış ise 1 dir 0 önce yazılmış ise 0 dır diyerek en sağdaki kodlamayı oluşturduk. İlk 2 sütunda sürelerin yaklaşık 900mikro saniye olduğu görülmektedir. RC5 sinyalinde LOW ve HIGH süreleri 889us dir. Fakat bunun yanı sıra 1700 ve 1800 gibi değerler göreceksiniz. Bunlar ise 2 sinyalin HIGH, HIGH veya LOW, LOW olması durumudur. Bu durumda olanları ise 3. Ve 4. Sütunda nasıl ayırdığıma dikkat edin. Çünkü HH sinyalinin ilk H seviyesi sol tarafındaki yarım periyot olan LOW seviyesine aittir yani sol taraftaki sinyalin bitişidir. 2. H sinyali ise sağ tarafındaki yarım periyot sinyalin başlangıç noktasıdır. Bu şekilde düşündüğümüzde 1800 gibi bir sinyal geldiğinde 1800-1 yazıldıktan sonra 1800-2 si ise hemen arkasından yazılarak sinyalin şekline uydurulmuş olacaktır. Aşağıda gördüğünüz gibi bit6 ve bit7 birleşiktir. Bu şekildeki sinyali yukarıda anlattığım gibi tabloya 1800-1yani bit6 mavi çizgi 1800-2 ise bit7 kırmızı çizgi olarak düşüneceğiz.



Bu şekilde RC5 sinyalini de çözmüş oluruz. Toplamda 14 bitten oluşur.   
START1+START2+TOGGLE+5 bit adres+6 bit komut şeklindedir.

Programda ise **void tekrarla()** kısmı 3. Ve 4. Satırları hazırlamaktadır. Bundan sonra ise **void sonkod()** kısmı ise en sağdaki sütunu oluşturur. Sonrasında ise kod içerisinden adres ve komut kısmını ayırarak yazar.

Programları birleştirme: Ben protokolleri anlatmak için programları ayrı ayrı yaptım. Bunları birleştirdiniz ve sinyal geldi hangisine göre kod çözeceksiniz. Bu da verdiğim programlar arasında protokol.ino programıyla anlaşılacaktır. Bu program ise gelen sinyali gördüğünde start palsine bakarız. Start palsi 9ms ise NEC protokolü deriz. 4,5ms ise samsung NEC diyebiliriz şayet start palsi 889ms civarıysa RC5 kodu diyerek programınızı yönlendirebilirsiniz. Bu program sadece start palsinin sürelerini pulsin komutuyla kontrol eder.

**Bütün bunları anlatırken bir noktayı anlatmadım.** **En önemli olan kısmı ise budur**. Başlangıç palsinin yakalanması için gelecek olan büyük palsin beklenmesi.   
İki kod bilgisi arasında oldukça uzun bir geçiş süresi var. NEC protokolünde, 108ms RC5 te ise 114ms de bir sinyal tekrarlanır. Diğer protokollerde de bunu göreceksiniz.  
Burada şunu unutmamak lazım. RC5 kodunda her tekrarda sinyal kendini toggle biti değişmeksizin tekrar eder. Birinci pals katarını kaçırsanız bile 2.sini mutlaka yakalayabilirsiniz. Bu da pulsin komutunda 80ms boyunca HIGH ya da LOW beklemekle olur. Bu süre sonunda PULSIN timeout verir ve 0 döndürür. Bu şekilde HIGH ya da LOW uzun bir aralık tespit edilir. Biliyoruz ki bu aralıktan sonra start palsi gelecek. Biz de kodu yakalayabileceğiz.

Fakat NEC protokolünde tuşa bastığımızda sinyal 1 kez tuş kodu yollar ve arkasından sadece tekrar sinyali olarak start palsini yollar. Bu şekilde düşünürsek öncelikli olarak ilk gelen palsi NEC ile yakalamak inceleyip NEC yada benzeri değilse RC5 olduğunu düşünüp tuş daha bırakılmadan uzun palsi bekleyip kodu tekrar almak bir çözüm olabilir. Tabii ki bu arada ilk yakaladığımız sinyalin start palsini incelememiz ve karar vermemiz de gerekebilir. Bu şekilde hangi kumanda protokolü olduğunu anlayabiliriz. Ona göre tuş bırakılmadan o protokole göre sinyali bir kez daha uzun sinyali bekleyip kayıt ederek kullanıcıya bilgiyi gönderebiliriz. Bunun dışında kendi Algoritmanızı da oluşturabilirsiniz.

**Rc5 tekrar ve NEC tekrar kısımlarını cursor hesaplamayla koy**

PIC işlemcisiyle nasıl yaparız derseniz programlar arasında aynı algoritmayı kullanan PBP ile gerçekleştirilmiş bir program var.   
Benzer algoritmayı kullanıp programları kendi işlemcinize veya kendi programlama dilinize uyarlayabilirsiniz. Kütüphane kullanılmadığından kolaylıkla adapte edebilirsiniz.   
Zor mu kolay mı derseniz bence kolay. Aşağıdaki satırlar RC5 sinyalini yakalamak için yeterli olmaktadır. Sonrası ise gelen sinyalin analiz edilmesidir. Analiz yöntemi yukarıda protokol anlatımlarında anlatıldığı gibi, sinyalin 1 veya 0 olduğuna göre ve LSB mi MSB mi önce gönderiliyor bunu bilmekle yapılır. Aşağıdaki program PIC 16F1827 ile yapılmış bir programın sinyal yakalaması. Önce start palsini sonra LOW seviyesi daha sonra ise HIGH seviyesi kontrol ederek süreleri diziye aktarır.

basla:  
 hserout ["IR sinyali gonder ",13,10]  
 temp=0:temp1=0  
 while temp1<14000  
 PULSIN RECV\_PIN,1,temp1  
 wend

for I=0 to 20  
PULSIN RECV\_PIN,0,sayL[I]  
Next

while temp1<14000  
PULSIN RECV\_PIN,1,temp1  
wend

Buraya kadar NEC + SAMSUNG + RC5 protokollerini anlatmaya çalıştım. Fakat diğer protokollere de baktığınızda, bu mantık ile onları da kolaylıkla çözebileceğinizi sanıyorum.

**Tüm programlar bu linktedir.** ( Programları yazıya eklemedim çünkü her biri yazıyı uzatacak ve okuyanı rahatsız edecekti. )

**Videolardan sonra** ise elimdeki kumandaların Alıcı gözden sonraki osilaskop görüntülerini inceleyebilirsiniz.

**NOT:** Sizlere kumandanın şu tuşu şu koddadır diyemediğim için örnek kod veremiyorum. Çünkü bazı tv modelleri 1 nolu tuşa 1 kodunu verirken bazıları farklı kod verebiliyor. Bundan dolayı sizi yanıltırım. RC5 te veya diğer protokollerde Hangi butona hangi kod verileceği yazılmış olsa da üreticiler buna %100 uymuyor diye gördüm. Bundan dolayı kendi kodunuzu kendiniz bulmalısınız. Sadece referans olarak size sinyal görüntülerini veriyorum. Elimdeki her kumandanın sinyalini osilaskoptan alıp yazıya ekledim.