

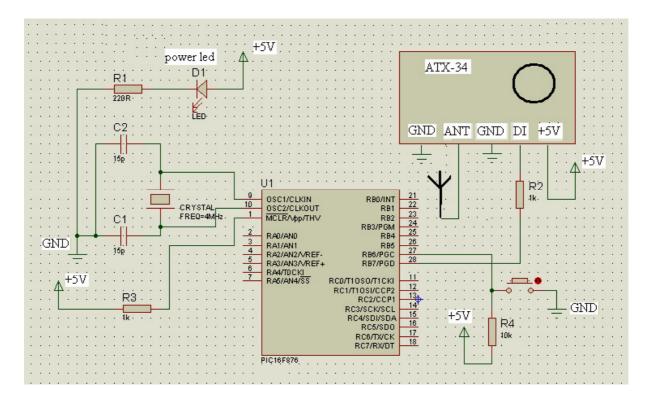
WIRELESS TECHNOLOGIES

UN-0805v01_ATX-ARX

ARX-34 ve ATX-34 KULLANARAK DATA GÖNDERMEK VE ALMAK İÇİN GEREKLİ YAZILIM ve DONANIM

WIRELESS COMMUNICATION PRODUCTS

VERICI DEVRESI

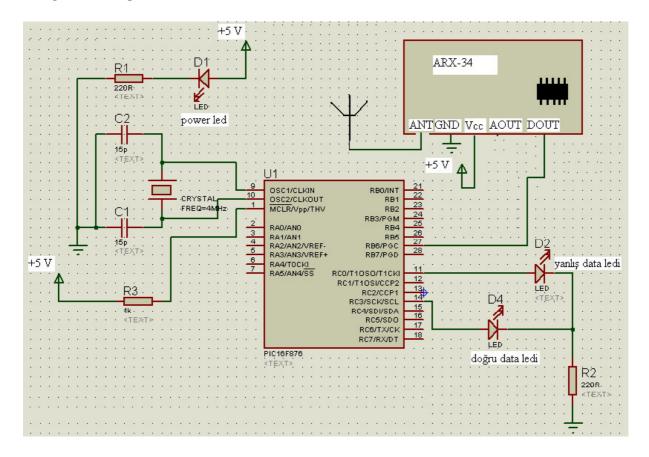


- Yazılım aşağıda verilmiştir. Yazılım PIC16F876 veya PIC16F876A için yazılmıştır.
- Baud rate 2400 dür.
- Yazılımda zamanlama çok önemlidir. Bu yüzden 4 Mhz kristal kullanılmalı veya yazılım kullanılacak kristal frekansına uyarlanmalıdır.
- Devre şemasının çizildiği Proteus6 ISIS programında PIC16F876 nın Vss ve Vdd bağlantıları belirtilmemiştir. Devre kurulurken bu bağlantılar unutulmamalıdır. Vss ye GND Vdd ye +5V DC bağlanmalıdır.

```
/* Uygulama Notu UN-0805v01_ATX-ARX
/* SOFTWARE
/* File: verici.c
/* Microcontroller: Microchip PIC16F876
/* Yazılımin içinde bir çok bekleme rutini kullanılmıstır.
/* Crystal frekansı ve compiler farklılıgından dogacak
/* degisiklikler kullanıcı tarafından dikkate alınmalıdır.
/* Compiler olarak HI-TECH Software kullanılmıştır.
/* XT=4MHz
/* Yazan: Mehmet Yağcı, UDEA
/* Contact: UDEA Elektronik 0.312.395 68 75
        info@udea.com.tr
#include <pic.h>
#include <delay.c>
#define DI RB7
                                         // RB7 pini kullanılarak data gönderiliyor
#define button RB6
                                         // gönderme yapmak için kullanılan tuş
int count=0;
                                         // 1 byte=8 bit
                                         // gönderilecek verilerin atandığı değişken
unsigned char data=0x00;
unsigned char data_set=0x00;
                                         // bit_reset ve bit_set denetim değişkeni
unsigned char mask=0x80;
void p_width()
DelayUs(200);
                           // baud rate değerimiz 2400, her bir pluse 416 Us olmalı
DelayUs(175);
                           // send(data) programındaki gecikmeler dikkate alındığında
                           // data gönderme sırasında pulse width 375 Us olur
}
void send(data)
                           // data gönderen alt program
       while(count<=7)</pre>
             data_set=data & mask;
             if(data_set==0x80)
              {
             DI=1;
              }
             else
             DI=0;
             p_width();
             if(count==7)
```

```
count++;
              break;
              data=data<<1;
              count++;
       }
       count=0;
       return;
}
void main()
                                            // ana program
TRISB=0X40;
                                            // portb nin tüm pinleri çıkış yapıldı
       for(;;)
       {
                                           // tuş basılı tutulduğu sürece data gönderir
              if(button==0)
              send(0x55);
                                           // 5 byte 0x55 (preamble)
              send(0x55);
              send(0x55);
              send(0x55);
              send(0x55);
              send(0x00);
                                            // 5 byte 0x00 ve 5 byte 0xff (senkron)
              send(0x00);
              send(0x00);
              send(0x00);
              send(0x00);
              send(0xff);
              send(0xff);
              send(0xff);
              send(0xff);
              send(0xff);
              DI=0;
                                                          // stop biti
              DI=1;
                                                          // start biti
              p_width();
              send(0x42);
                                                          // gönderilen data
              DelayMs(250);
                                                          // 500 ms gecikme
              DelayMs(250);
               }
       }
}
```

ALICI DEVRESİ



ARX-34 içerisinde bir voltaj regülatorü bulunmamaktadir. Tasarim pil kullanimi düsünülerek yapilmistir. Bu nedenle besleme voltajinda belirtilen degerlere dikkat edilmelidir.

Modül belirtilen degerlerin altında bir besleme yapildiginda kararsız çalisacaktır. Besleme voltaji +5 VDC ve tapraklama GND baglantisi belirtilen degerlerin üzerinde veya ters olursa, modülde kalici tahribatlara yolaçilabilir.

Modül basit bir anten baglanti pinine sahiptir. Uygun bir UHF anten dogrudan bu pine baglanabilir. ARX-34 modülüne baglanabilecek en basit anten 17.3cm uzunlugundaki bir kablonun anten girisine lehimlenmesidir. Anteni, modülden uzak bir yere baglamaniz gerekiyorsa 50 Ohm Coax anten kablosu kullanmaniz gerekmektedir. Anten kablosunun topraklamasi, modulun anten girisine yakin bir yerden yapilmalidir.

Asagidaki kurallar size yardimci olacaktir:

- * Anten 50 ohm empedancli olmalidir.
- * Lambda/4 whip anten boyu 433MHz. için yaklasik 17.3 cm dir.
- * Anteni modül dik gelecek sekilde yukariya veya asagiya dogru monte ediniz.
- * Anteni metal bir hazne içine koymayiniz.
- * Insan vücudu metal objeler gibi etkiler gösterebilirler. Tasinabilir alici veya vericiler vücuttan uzak bir sekilde elde tutulmalidir.

- * En iyi iletisim mesafesi her iki taraftaki antenlerin birbirini görmesi ile elde edilebilir. Herhangi bir obje veya metal bir engel iletisim mesafesini düsürecektir.
- * Sinyal göndermeleri, gönderilen sinyallerin metal yüzeylerden, binalardan vb. gelen yansimalardan etkilenirler. Bu yanlis data alimlarina yolaçabilir.

Aşağıdaki yazılımla bilgisayardan yollanan datanın sencron bilgisi yakalanıp sonraki karakterler LCD'ye yazdırılmaktadır. Mesaj sonu bilgisi (bu yazılımda nokta karakteri) gelene kadar her karakter LCD'ye yazdırılır. Mesaj sonu bilgisi geldiğinde gönderilen mesaj 3 saniye kadar ekranda bekletilir. Daha sonra yeni bir mesaj için LCD temizlenir ve mikroişlemci sencron aramaya devam eder.

```
/* Uygulama Notu UN-0805v01_ATX-ARX
/* SOFTWARE
/* File: alıcı.c
/* Microcontroller: Microchip PIC16F876
/* Yazılımin içinde bir çok bekleme rutini kullanılmıstır.
/* Crystal frekansı ve compiler farklılıgından dogacak
/* degisiklikler kullanıcı tarafından dikkate alınmalıdır.
/* Compiler olarak HI-TECH Software kullanılmıştır.
/* XT=4MHz
/* Yazan: Mehmet Yağcı, UDEA
/* Contact: UDEA Elektronik 0.312.395 68 75
            info@udea.com.tr
#include
            <pic.h>
#include
             <delay.c>
#define Dout RB6
                                              // ARX den datanın alındığı pin
                                              // datanın yanlış okunduğunu gösteren led
#define LED RC0
#define DLED RC3
                                              // datanın doğru okunduğunu gösteren led
#define bit_set(var,bitno)
                          ((var) |= (1 << (bitno)))
                                                     // istenilen biti 1 yapan macro
                                                     // istenilen biti 0 yapan macro
#define bit_reset(var,bitno) ((var) &= \sim(1<<(bitno)))
int plus0,plus1,plus2;
                                        // savıcılar
                                        // data bit sayıcısı
int count, say;
unsigned char data=0x00;
                                        // datanın atanacağı değişken
void led();
                                 // data yanlış okundu ledini yakan alt program
void Dled();
                                 // data doğru okundu ledini yakan alt program
                                 // senkron 5 bit 0x00 1 okuyan alt program
void fivebit0();
                                 // senkron 5 bit 0xff i okuyan alt program
void fivebit1();
void data_oku();
                                 // alınan datayı okuyan program
```

```
//----ana program-----
void main()
TRISB=0XFF;
                     //PORTB input tanımlandı
TRISC=0X00;
                     //PORTC output tanımlandı
PORTB=0;
PORTC=0;
     for(;;)
     fivebit0();
}
//----
void led()
     LED=1;
     DelayMs(250);
     DelayMs(250);
     DelayMs(250);
     DelayMs(250);
     LED=0;
     DelayMs(250);
     DelayMs(250);
     plus0=0;
     plus1=0;
     plus2=0;
     say=0;
     data=0;
     return;
//-----
void Dled()
{
     DLED=1;
     DelayMs(250);
     DelayMs(250);
     DelayMs(250);
     DelayMs(250);
     DLED=0;
     DelayMs(250);
     DelayMs(250);
     plus0=0;
     plus1=0;
     plus2=0;
     say=0;
     data=0;
     return;
}
//----5 bit 0x00 arayan program-----
```

```
void fivebit0()
      for(;;)
       {
             if(Dout==0)
                                                //senkron 5 bit 0x00 aranıyor
                    for(;;)
                    DelayUs(1);
                    plus0++;
                                                // 5 byte 0x00 bulundu
                           if(plus0>=600)
                           while(Dout==0);
                           fivebit1();
                                                // 5 bte 0xff arayan programa git
                           return;
                           if(Dout==1)
                                                // 5 byte 0x00 bulunamadı
                           break;
                    plus0=0;
                                        // 0x00 yeterli uzunluğu sağlamıyor başa dön ara
              }
       }
    -----5 bit 0xff arayan program------
void fivebit1()
      for(;;)
                                         //senkron 5 byte 0xff araniyor
             if(Dout==1)
                    for(;;)
                    DelayUs(10);
                    plus1++;
                           if(plus1>=600)
                                           // 5 bit 0xff bulundu
                           while(Dout==1);
                           data_oku(); // senkron yakalandı data_oku programına git
                           return;
                           }
                           if(Dout==0)
                                                       // 5 byte ff bulunamadı
                                  plus0=0; // 0xff yeterli uzunluğu sağlamıyor başa dön ara
                                  plus 1=0;
                                  return;
                           }
                    }
              }
       }
}
```

```
void data_oku()
                               // 8 bit data oku
      while(say<=7)</pre>
      if(Dout==1)
      bit_set(data,0);
                                      // biti 1 yap
      else
      bit_reset(data,0);
                                      // biti 0 yap
      if(say==7)
      break;
      say++;
      data=data<<1;
                                      // datayı bir bit sola kaydır
      DelayUs(200);
      DelayUs(210);
      if(data==0x42)
                                      // 8 bit okundu alınan datayı kontrol et
      Dled();
                                            // data doğru
      return;
      }
      else
      {
      led();
                                            // data yanlış
      data=0;
      say=0;
      return;
                                            // ana programa dön
      }
//-----
```