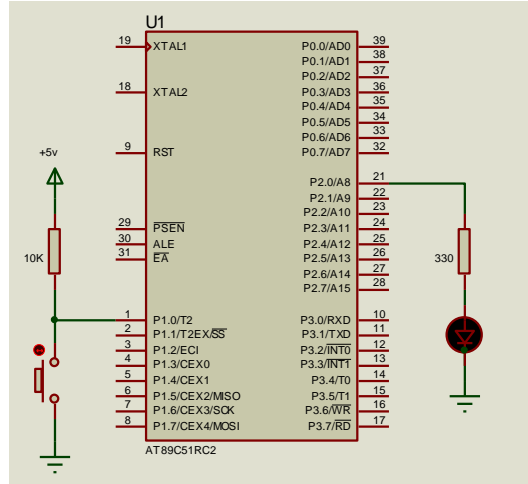


➤ Giriş/Çıkış İşlemleri (Buton ile LED Kontrol Uygulaması)

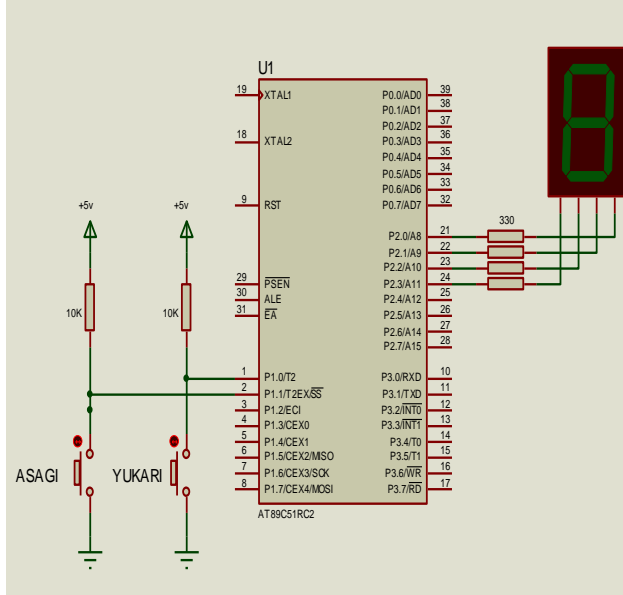


```

ORG 0H
SJMP BASLA
ORG 30H
BASLA:MOV P1,#0FFH      ;P1 portu giriş olarak ayarlandı
      MOV P2,#0         ;Led başlangıçta yanmıyor
BAS:  JB P1.0,BAS       ;P1_0'a bağlı Buton kontrol ediliyor Basıldı mı?
BIRAK:JNB P1.0,BIRAK    ;Buton bırakıldı mı?
      CPL P2.0          ;LED'i tersle (toggle)
      SJMP BAS          ;Butonu tekrar kontrol etmek için Bas'a dallan
END

```

➤ Giriş/Çıkış İşlemleri (Aşağı/Yukarı BCD Sayıcı Uygulaması)



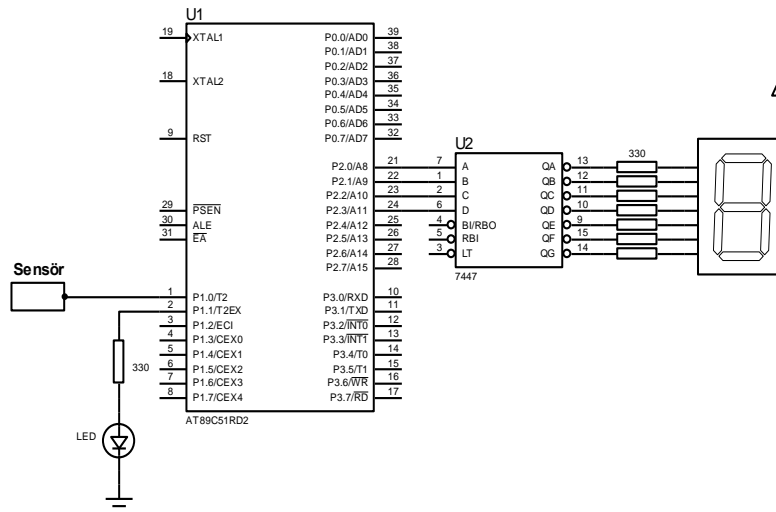
```

ORG 0H
SJMP BASLA
ORG 30H
BASLA: MOV P1,#0FFH
      ;P1'i giriş kur
      CLR A
      MOV P2,A
YUKARI: JB P1.0,ASAGI
      ;Yukarı butonu
YUKARI2:JNB P1.0,YUKARI2
      CJNE A,#9,ARTTIR
      MOV A,#0
      MOV P2,A
      SJMP ASAGI
ARTTIR: INC A
      MOV P2,A
ASAGI:  JB P1.1,YUKARI
      ;Aşağı butonu
ASAGI2: JNB P1.1,ASAGI2
      CJNE A,#0,AZALT
      MOV A,#9
      MOV P2,A
      SJMP YUKARI
AZALT:  DEC A
      MOV P2,A
      SJMP YUKARI
END

```

➤ Giriş/Çıkış İşlemleri

Bir fabrika ortamında yürüyen bantlar üzerinden geçen ürünlerin sayılması ve her 10 üründe bir paketleme işleminin yapılması amaçlanmaktadır. 8051'in P1.0 ucuna bağlı olan bir algılayıcı her ürün geçişinde düşen kenarlı bir sinyal oluşturmaktadır. Algılayıcıdan geçen her üründe P2 portuna bağlı olan 7 segmentli gösterge 1 artmaktadır. 10 ürün tamamlandıktan sonra paketleme işlemi gerçekleştiğinde 7 segmentli gösterge 0'lanmakta ve paketleme işleminin tamamlandığını göstermek üzere P1.1 ucuna bağlı olan LED yanıp sönmektedir. Gerekli assembly programını yazınız?



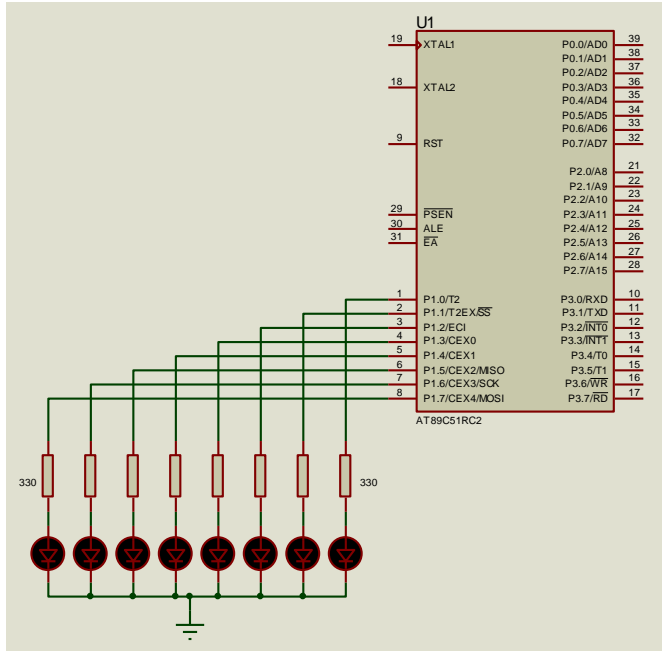
```

        ORG    0H
        SJMP   BASLA
        ORG    30H
BASLA:  SETB   P1.0           ;P1.0 sensör girişi olarak ayarlandı
        CLR    P1.1         ;Led başlangıçta yanmıyor
        CLR    A             ;ürün sayısı aküde tutulacak

Kontrol:MOV P2,A
Sensor: JB P1.0,Sensor      ;ürün geçti mi?
Sensor1:JNB P1.0,Sensor1
        INC    A             ; ürün geçti 1 arttır
        CJNE   A,#10,Kontrol
        CLR    A             ;paketleme tamam aküyü sıfırla
        SETB   P1.1
Bekle:  DJNZ   R0, Bekle
        DJNZ   R1,Bekle
        CLR    P1.1
        SJMP   Kontrol
        END

```

➤ Yürüyen ışık uygulaması

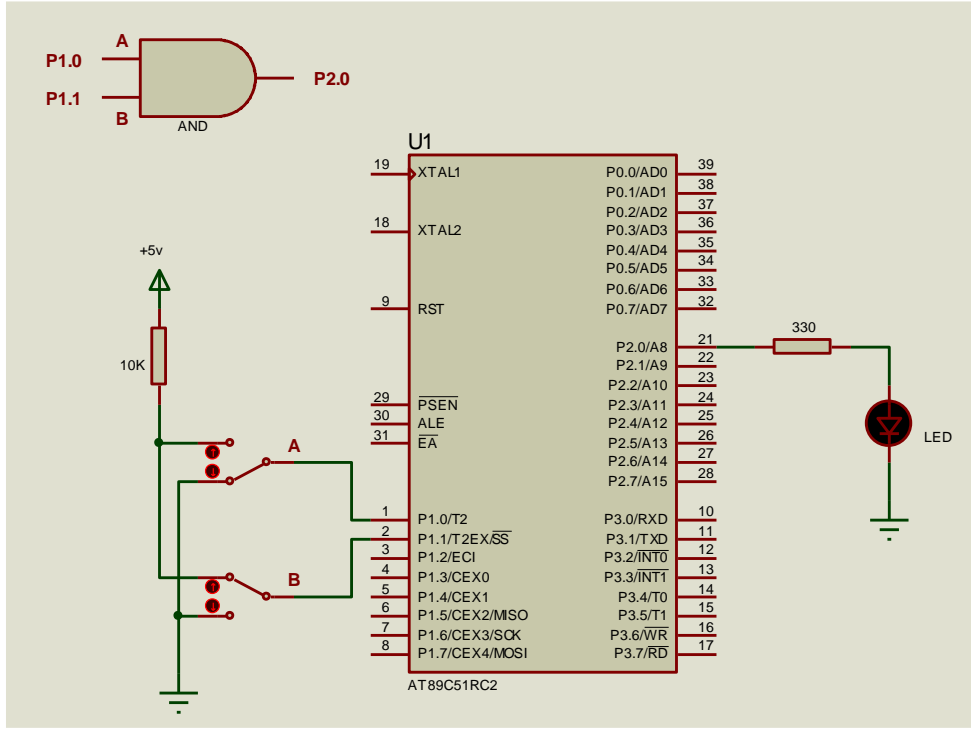


```

ORG 0H
SJMP BASLA
ORG 30H
BASLA:MOV A,#80H          ;başlangıç değerini yükle en soldaki led 1
SAG:  MOV P1,A            ;aküyü P1 portuna yükle
      CALL GECIKME        ;gecikme alt programını çağır
      RR A                ;aküdeki değeri 1 bit sağa döndür
      CJNE A,#01h,SAG     ;P1_0 1 olana kadar (en sağ bit) SAG'a dallan
SOL:  MOV P1,A            ;aküyü P1 portuna yükle
      CALL GECIKME        ;gecikme alt programını çağır
      RL A                ;aküdeki değeri 1 bit sola döndür
      CJNE A,#80h,SOL     ;P1_7 1 olana kadar (en sol bit) SOL'a dallan
      SJMP SAG            ;SAG etiketine dallan
GECIKME:MOV R0,#255
      MOV R1,#255
BEKLE:DJNZ R0,BEKLE       ;R0'ı 1 azalt '0' değilse Bekle'ye dallan
      MOV R0,#255
      DJNZ R1,BEKLE       ;R1'i 1 azalt '0' değilse Bekle'ye dallan
      RET                ;alt programdan dön
      END

```

➤ Giriş/Çıkış İşlemleri (VE Kapı Uygulaması)



❖ Çözüm 1 (İstenilen Çözüm)

```

ORG 0H
SJMP LOOP
ORG 30H
LOOP: MOV C, P1.0      ;Ve kapısının A girişini Carry Flage yükle
      ANL C, P1.1      ;A girişi ile B girişini Ve işlemine tabi tut
      MOV P2.0,C       ;Ve işleminin sonucunu LED'te göster
      SJMP LOOP
END

```

Not: C (Carry Flag) yerine herhangi bit değişkeni veya bit adreslenebilir bir kayıtçı da kullanılabilir.

❖ Çözüm 2

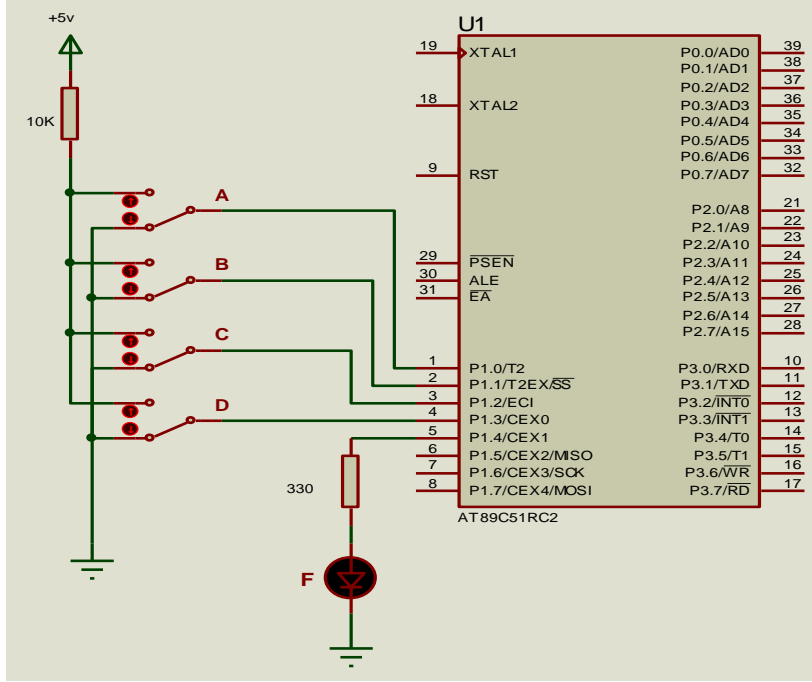
```

ORG 0H
SJMP BASLA
ORG 30H
BASLA: JB P1.0,KONTROL ;P1_0=1 ise P1_1'i kontrol et
      CLR P2.0         ;P2_0 ucuna bağlı LEDi söndür
      SJMP BASLA
KONTROL: JB P1.1,YAK   ;P1_1=1 ise YAK'a git
      CLR P2.0         ;P2_0 ucuna bağlı LEDi söndür
      SJMP BASLA
YAK:   SETB P2.0       ;P2_0 ucuna bağlı LEDi yak
      SJMP BASLA
END

```

➤ Giriş/Çıkış İşlemleri (Lojik Fonksiyon Gerçekleştirilmesi)

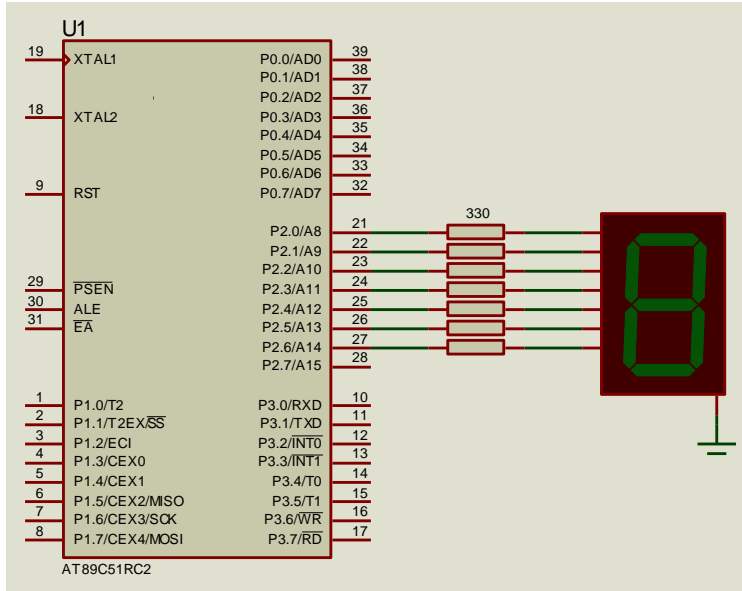
8051'in P1 portunu $F = (A+B.D) \cdot C$ fonksiyonunu gerçekleştirecek şekilde programlayınız



```
ORG 0H
SJMP LOOP
ORG 30H
LOOP: MOV C, P1.1
      ANL C, P1.3
      ORL C, P1.0
      ANL C, P1.2
      MOV P1.4,C
      SJMP LOOP
```

➤ Sıralı (index) Adresleme

Sayfa 270- 7 Parçalı Gösterge Uygulaması 2'nin assembly dilinde yazılmış halidir.



P2 portuna doğrudan bağlı olan Ortak Katotlu 7 Segment Displayde 0'dan F'e kadar olan sayıları gösteren program. 0-F değerlerinin karşılıkları Tablo isimli dizide DB ile tanımlanmıştır.

- **ORG Adres** talimat (yönerge) satırını kullanarak Tablo etiketinde belirtilen değerleri belirli bir bellek adresinden itibaren belleğe yerleştiriniz.
- CLR A satırının altına **MOV DPTR,#TabloAdres-1** komut satırı ilave edilmelidir. Böylelikle tablo dizisinin bellekteki başlangıç adresinden bir öncesi belirtilmektedir.
- MOVC A,@A+PC satırı yerine **MOV A,@A+DPTR** komut satırı yazılmalıdır.

```

    ORG 0H
    SJMP BASLA
    ORG 30H
BASLA: CLR A          ;aküyü sıfırla
        MOV DPTR,#005FH
;Tablo dizisinin başlangıç adresinin bir öncesi (60-1=5F)
CALIS: CALL SAY        ;say alt programını çağır
        MOV P2,A        ;aküdeki değeri P2'ye aktar
        MOV A,R2        ;R2'deki değeri (Tablo dizisinin indisi) aküye aktar
        CALL GECIKME    ;gecikme alt programını çağır
        CJNE A,#16,CALIS ;0'dan F'e kadar tüm rakamlar yandı mı
        SJMP BASLA      ;Basla etiketine dallan
GECIKME: MOV R0,#255
        MOV R1,#255
BEKLE: DJNZ R0,BEKLE    ;R0'ı 1 azalt '0' değilse Bekle'ye dallan
        MOV R0,#255
        DJNZ R1,BEKLE    ;R1'i 1 azalt '0' değilse Bekle'ye dallan
        RET              ;alt programdan dön
SAY: INC A              ;aküdeki değeri 1 arttır (Tablo dizisinin indis değeri)
        MOV R2,A          ;Tablo dizisinin indis değerini R2'ye yükle
        MOVC A,@A+DPTR
;akü ile DPTR'yi topla ilgili adresteki bilgiyi aküye yükle
        RET
        ORG 60h          ;Tablo dizisini belleğe 60h adresinden itibaren yerleştir
;0'dan F'e kadar rakamların 7Segment karşılıkları
TABLO: DB 3FH, 06H, 5BH, 4FH
        DB 66H, 6DH, 7DH, 07H
        DB 7FH, 6FH, 77H, 7CH
        DB 39H, 5EH, 79H, 71H
        END

```