14. MİKROİŞLEMCİ PROGRAMLAMA TEKNİKLERİ

14.1. Programlamaya Giriş

Örnek Pr. 14-1 Bellekte bir adreste bulunan verinin başka bir adrese transfer edilmesi.

a) 0040H bellek adresindeki 8-bit veriyi 0042H adresine transfer eden programı yazınız.

b) 0040H bellek adresindeki 16-bit veriyi 0042H adresine transfer eden programı yazınız.

Çözüm:

a)

0040H adresindeki 8-Bit veriyi A yazmacına yükle	LDAA 0040H
A yazmacındaki 8-Bit veriyi 0042H adresine sakla	STAA 0042H

b)

0040H adresindeki 16-Bit veriyi X yazmacına yükle	LDX 0040H
X yazmacındaki 16-Bit veriyi 0042H adresine sakla	STX 0042H

Örnek Pr. 14-2 0040h bellek adresi ile 0041h adresindeki 8-bit veriyi toplayan ve sonucu 0042h adresinde saklayan programı yazınız.

Çözüm:

Toplamada başlangıç için elde sıfır yapılır	CLC
0040H adresindeki veriyi A yazmacına yükle	LDAA 0040H
A yazmacındaki veriye 41H adresindeki veriyi ekle	ADCA 0041H
A yazmacındaki veriyi 0042H adresinde sakla	STAA 0042H

Örnek Pr. 14-3 0040h bellek adresindeki 8-bit veriden 0041h adresindeki 8-bit veriyi çıkaran ve sonucu 0042h adresinde saklayan programı yazınız.

Çözüm:

Çıkarmada başlangıç için ödünç(elde) sıfır yapılır	CLC
0040H adresindeki veriyi A yazmacına yükle	LDAA 0040H
A yazm.daki veriden 41H adresindeki veriyi çıkar	SBCA 0041H
A yazmacındaki veriyi 0042H adresinde sakla	STAA 0042H

Örnek Pr. 14-4 0040h bellek adresindeki 8-bit veriyi 1-bit sola öteleyen ve sonucu 0041h adresinde saklayan programı yazınız.

Çözüm:

0040H adresindeki veriyi A yazmacına Yükle	LDAA 0040H
A yazmacındaki veriyi 1-bit sola ötele	ASLA
A yazmacındaki veriyi 0041H adresinde sakla	STAA 0041H

Örnek Pr. 14-5 0040h bellek adresindeki 8-bit verinin düşük ağırlıklı 4-bitini 0041h adresinde saklayan programı yazınız. 0041h bellek adresindeki 8-bit verinin yüksek ağırlıklı 4-bitini sıfırlayın.

Çözüm:

0040H adresindeki veriyi A yazmacına yükle	LDAA 0040H
A yazm. veri ile 00001111 değerini VE işlemi yap	ANDA #00001111B
A yazmacındaki veriyi 0041H adresinde sakla	STAA 0041H

Örnek Pr. 14-6 0040h bellek adresindeki veriyi sıfır ile dolduran (temizleyen) programı yazınız.

Çözüm:

A yazmacına 0 değerini yükle	LDAA #0H
A yazmacındaki veriyi 0040H adresinde sakla	STAA 0040H
veya	
0040H adresindeki veriyi sıfırla	CLR 0040H

14.2. Mikroişlemcilerin Gelişmiş Komutları

Mikroişlemcinin bazı komutları program tasarımında sıkça karşılaşılacağı düşünülen bir fonksiyonu yerine getirir.

Örnek Pr. 14-7 32+29 İşlemini Toplamadan Sonra Akümülatörü Ondalığa Ayarla (DAA) Komutunu kullanarak yapınız.

Çözüm:

Çevirici giriş kaynak dosyası:

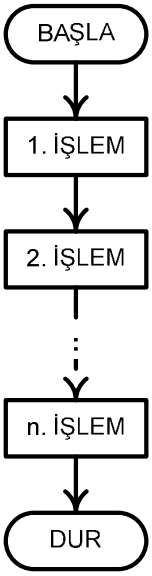
```
; ********************************************************************************
; *  BCD kodlu 2 basamak onaltılık sayıların toplamını DAA komutunu      *
; *  kullanarak bulan program                                            *
; ********************************************************************************
;
            ORG 0D019H      ; programın başlangıç adresi
BASLA:      CLC             ; Elde bayrağını sıfırla
            LDAA  #32H      ; A akümülatörüne 32H yükle
            ADCA  #29H      ; A aküm. Elde ile 29h topla
            DAA             ; sonucu BCD'ye dönüştür
            END             ; programın sonu
```

Çevirici program listesi çıkış dosyası:

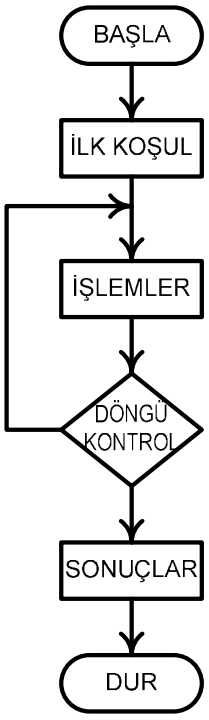
```
D019                  ORG   0D019H  ; programın başlangıç adresi
D019  0C      BASLA:  CLC           ; Elde bayrağını sıfırla
D01A  8632            LDAA  #32H    ; A akümülatörüne 32H yükle
D01C  8929            ADCA  #29H    ; A aküm. Elde ile 29h topla=5Bh
D01E  19              DAA           ; sonucu BCD'ye dönüştür. 5Bh → DAA → 61h
0000                  END           ; programın sonu
```

14.3. Programlama İçin Akış Diyagramı Yöntemi

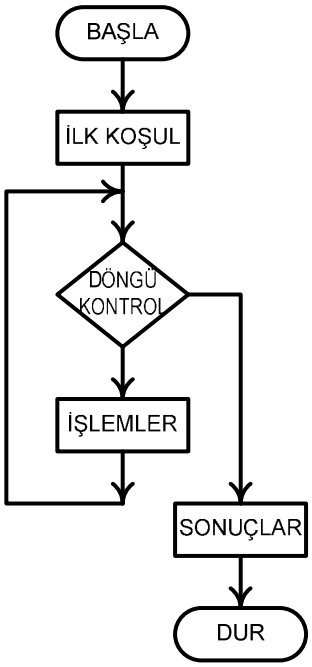
Program tasarımlarında çok karşılaşılan problemlerin genel olarak çözüm yöntemini veren akış diyagramı yapıları vardır.



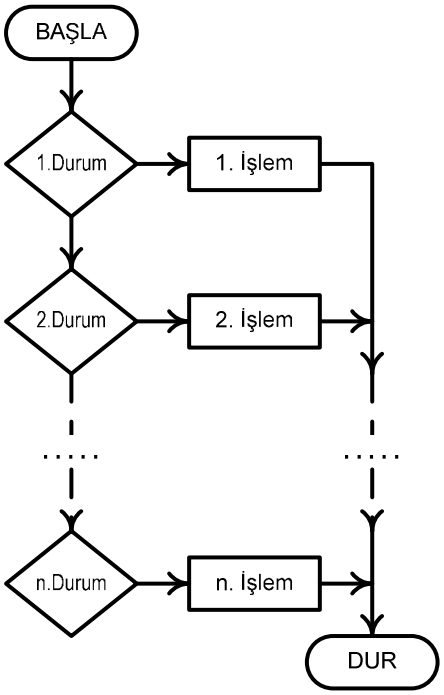
Şekil 14-1 Sıralı yapılan basit işlemler için akış diyagramı



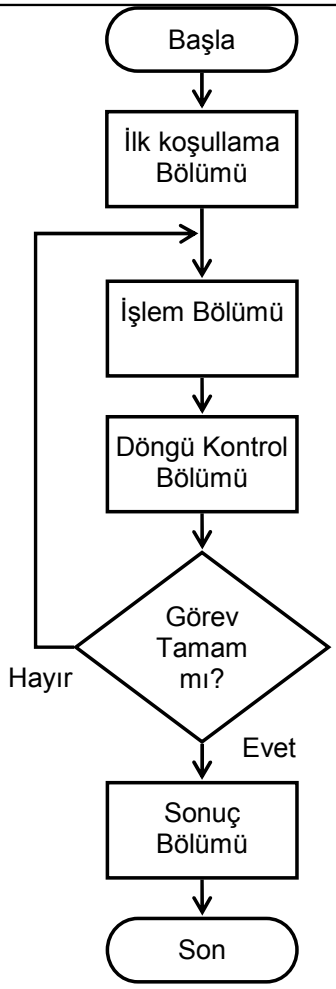
Şekil 14-2 Karar verme, tekrarlı işlemler gerektiren problemlerin çözümü için akış diyagramı



Şekil 14-3 Karar verme, tekrarlı işlemler gerektiren problemlerin çözümü için döngü çıkış kontrolünü önce yapan akış diyagramı



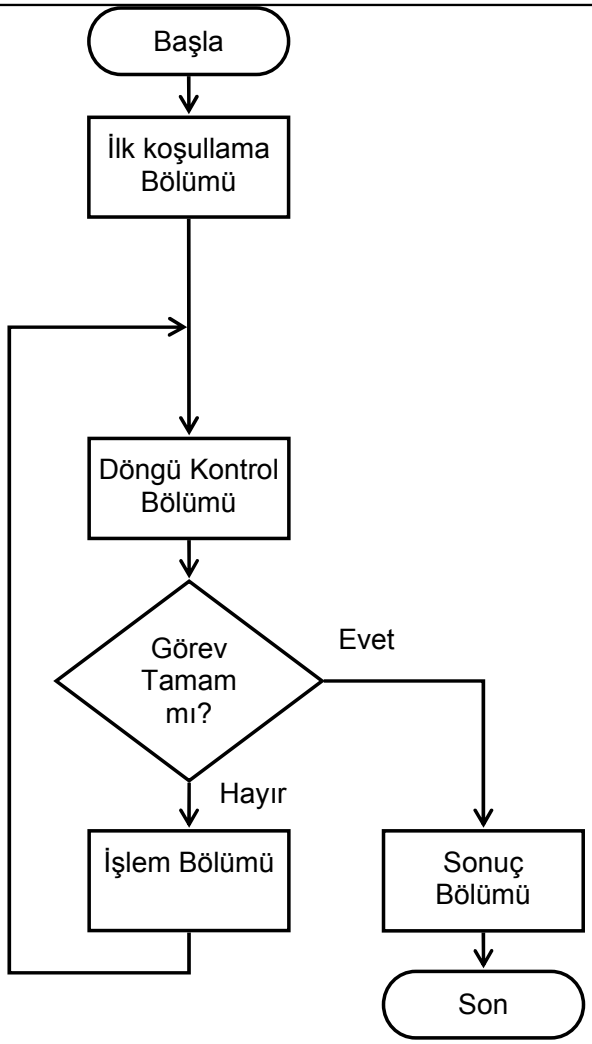
Şekil 14-4 Birden fazla karar verme, çok seçimli işlemler gerektiren problemlerin çözümü için akış diyagramı



Şekil 14-5 Program döngüleri için akış diyagramları

a) son işlemden sonra kontrol

b) son işlemden önce kontrol



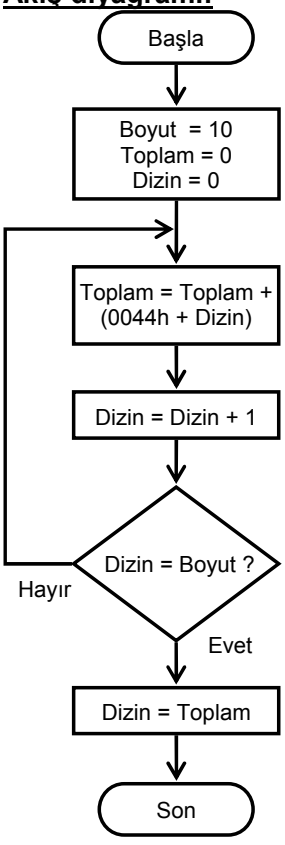
Örnek Pr. 14-8 Sayı dizisinin toplamının hesaplayan programın akış diyagramını çiziniz. 0040h bellek adresinde sayı dizisinin boyutunu, 0042h bellek adresinde sonucu ve 0044h bellek adresinden başlayarak sayı dizisini saklayınız. Ayrıca programın sonunda toplama sonucunu X yazmacında saklayınız.

Çözüm:
Programın akış diyagramı Şekil 14-6'da verilen şekilde tasarlanmıştır. Akış diyagramının sağında ise her bir bloğun gerçekleştirilmesi için 6800 mikroişlemcisi komut veya komut grubu gösterilmiştir.

Sayı dizisinin boyutu, BOYUT 0040H adresinde saklanmıştır. DIZI_ADR 0044H olarak verilmiştir.
Toplama sonucu 16-Bit alınırsa 42H:43H adresleri kullanılır.
TOPLAM =42H ve TOPLAM+1=43H olur.
İlk işlem (TOPLAM:TOPLAM+1)=(42H:43H)=0
A akümülatörüne, önceki toplama sonucu "LDAA TOPLAM+1" komutuyla yüklenir.
(X=0,X+44H=0044H) olarak, Akümülatörde toplanır. "TOPLAM+1" adresinde saklanır.
Her bir ara toplamda, 8-Bit toplamadan üreyen elde, 16-Bit toplama sonucunun yüksek ağırlıklı kısmı TOPLAM adresindeki değer artırılarak toplama sonucuna eklenir.
X dizin yazmacı artırılarak sonraki 8-Bit değerler
(X=1,X+44H=0045H),
(X=2,X+44H=0046H),
…
(X=9,X+44H=004DH) toplanır. İşlemler döngü içinde bu şekilde tekrarlanır.
Bu işlem X dizin yazmacı karşılaştırılarak BOYUT kadar tekrarlanır.
TOPLAM adresindeki 16-Bit sonuç X dizin yazmaca yüklenir ve program sonlandırılır.

(TOPLAM) (TOPLAM+1)
+ (X+44H)
(TOPLAM+1)
Eğer elde varsa
(TOPLAM)+1→(TOPLAM)

Akış diyagramı:



Program Karşılığı:

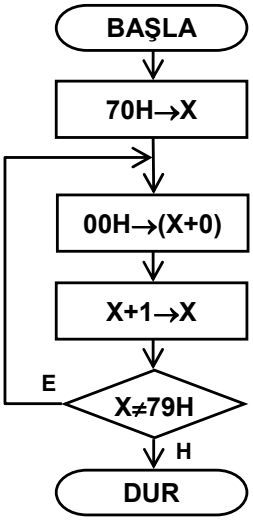
```
BOYUT    EQU   0040H
TOPLAM   EQU   0042H
DIZI_ADR EQU   0044H

         LDX   #10
         STX   BOYUT
         LDX   #0
         STX   TOPLAM

TOP1:    LDAA  TOPLAM+1
         ADDA  DIZI_ADR,X
         BCC   ELD_YOK
         INC   TOPLAM
ELD_YOK: STAA  TOPLAM+1

         INX

         CPX   BOYUT
         BNE   TOP1

         LDX   TOPLAM

         END
```

Şekil 14-6 Boyutu değiştirilebilen 10 elemanlı sayı dizisinin toplamı için akış diyagramı

Örnek Pr. 14-9 Bellekteki 70H ile 78H adresleri arasındaki alandaki baytların sıfır ile doldurulması (temizlenmesi).

Çözüm:

Akış diyagramı:



Program Karşılığı:

```
;programın başlangıcı
ORG     2000H
;başlangıç adres değerini X yazmacına yükle
LDX     #70H
;programdaki döngünün dallanma etiketi
L1:
;dizinlenmiş adresleme ile X+0 adresinin içeriğini sıfırla
CLR     0,X
;adres değerinin artırılması
INX
;son adres sıfırlandı mı?
CPX     #79H
BNE     L1
;programın sonu
END
```

```
                ; CLRMEM1.ASM
                ; 0070H→0078H adresleri arasındaki bellek alanındaki baytların
                ; sıfır ile doldurulması (temizlenmesi).
0000                    CPU   "6800.TBL"
0000                    HOF   "MOT8"
2000                    ORG   2000H     ; programın başlangıcı
2000 CE0070       (3)   LDX   #70H      ; başlangıç adres değerini X yazmacına yükle
2003 6F00   L1:   (7)   CLR   0,X       ; X+0 adresinin içeriğini sıfırla
2005 08           (4)   INX             ; adres değerinin artırılması
2006 8C0079       (3)   CPX   #79H      ; son adres sıfırlandı mı?
2009 26F8         (4)   BNE   L1        ; gelinmediyse L1'e git
0000                    END             ; programın sonu
```

Programın Analiz Tablosu:

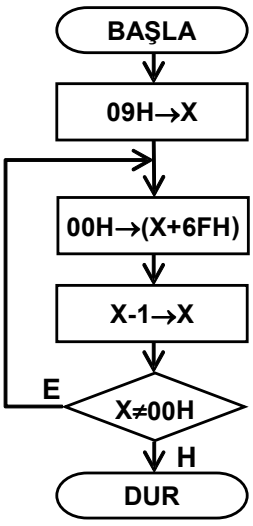
	PC	X	Z	(70H)	(71H)	(72H)	(73H)	(74H)	(75H)	(76H)	(77H)	(78H)
0	2000	?	?	?	?	?	?	?	?	?	?	?
1	2003	0070	0									
2	2009	0071	0	00								
3	2009	0072	0		00							
4	2009	0073	0			00						
5	2009	0074	0				00					
6	2009	0075	0					00				
7	2009	0076	0						00			
8	2009	0077	0							00		
9	2009	0078	0								00	
10	200B	0079	1									00

Programın toplam çalışma süresi = 3 + 9 * (7+4+3+4) = 3 + 9 * 18 = **165 sistem saati.**

Örnek Pr. 14-10 Bellekteki 70H ile 78H adresleri arasındaki baytların temizlenmesini, sıfır ile doldurulmasını sağlayan programın azalan adres sayacı ile tasarlanması.

Çözüm:

Akış diyagramı:



Program Karşılığı:

```
;programın başlangıcı
ORG     3000H
;sıfırla doldurulacak bayt adedini X yazmacına yükle
LDX     #9H
;programdaki döngünün dallanma etiketi
L1:
;dizinlenmiş adresleme ile X+6FH adresinin içeriğini sıfırla
CLR     6FH,X
;adres değerinin azaltılması
DEX
;son adres sıfırlandı mı?
BNE     L1
;programın sonu
END
```

Program Listesi Çıkış Dosyası:

```
                    ; CLRMEM2.ASM
                    ; 0078H→0070H adresleri arasındaki baytların
                    ; sıfır ile doldurulması (temizlenmesi).
0000                      CPU   "6800.TBL"
0000                      HOF   "MOT8"
3000                      ORG   3000H  ; programın başlangıcı
3000 CE0009         (3)   LDX   #9H    ; sıfırlanacak bayt sayısını X yazmacına yükle
3003 6F6F     L1:   (7)   CLR   6FH,X  ; X+6F adresinin içeriğini sıfırla
3005 09             (4)   DEX          ; bayt sayısını azalt
3006 26FB           (4)   BNE   L1     ; bayt sayısı sıfır değilse L1'e git
0000                      END          ; programın sonu
```

Programın Analiz Tablosu:

	PC	X	Z	(78H)	(77H)	(76H)	(75H)	(74H)	(73H)	(72H)	(71H)	(70H)
0	3000	?	?	?	?	?	?	?	?	?	?	?
1	3003	0009	0									
2	3006	0008	0	00								
3	3006	0007	0		00							
4	3006	0006	0			00						
5	3006	0005	0				00					
6	3006	0004	0					00				
7	3006	0003	0						00			
8	3006	0002	0							00		
9	3006	0001	0								00	
10	3008	0000	1									00

Programın toplam çalışma süresi = 3 + 9 * (7+4+4)
= 3 + 9 * 15 = **138 sistem saati.**

Örnek Pr. 14-11 Aşağıda verilen 6800 makine dili programın eksiklerini tamamlayınız ve her satırındaki komutun çalışma süresini ve açıklamasını yanına yazınız. PC program sayıcısı, X dizin yazmacı, A akümülatörü, durum yazmacının Z (sıfır) biti ve etkilenen bellek gözleri üzerinde analizini yapınız ve programın toplam çalışma süresini hesaplayınız.

Yazmaçların ilk durumu :

PC=E000H SP=006FH X=0002H A=25H B=FAH CCR=CCH

Bellek gözlerinin durumu (Bütün değerler onaltılık olarak verilmiştir!):

Adres	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0030	FC	FD	FE	FF	01	02	03	04	05	06	07	08	09	10	11	12
0040	13	14	15	16	17	18	76	91	A1	87	0B	17	43	B7	55	E8

```
E000                ORG   0E000H
E000 ____           LDX   #30H
E003 ____           STX   40H
E005 ____           LDX   #38H
E008 ____           STX   42H
E00A ____     L1:   LDX   40H
E00C ____           LDAA  0,X
E00E ____           INX
E00F ____           STX   40H
E011 ____           LDX   42H
E013 ____           STAA  0,X
E015 ____           INX
E016 ____           STX   42H
E018 ____           CPX   #39H
E01B ____           BNE   L1
E01D ____           NOP
0000                END
```

Çözüm: Programın tamamı:

```
E000                ORG 0E000h    ; programın başlangıç adresi 0E000h
E000 CE0030         LDX   #30h ;(3) X dizin yazmacına 30h değerini yükle.
E003 DF40           STX   40h  ;(5) X dizin yazmacını 40h:41h adresine yükle.
E005 CE0038         LDX   #38h ;(3) X dizin yazmacına 38h değerini yükle.
E008 DF42           STX   42h  ;(5) X dizin yazmacını 42h:43h adresine yükle.
E00A DE40     L1:   LDX   40h  ;(4) X dizin yazmacına 40h:41h adresindeki veriyi yükl
E00C A600           LDAA  0,X  ;(5) A akümülatörüne (X+0) adresindeki veriyi yükle.
E00E 08             INX        ;(4) X dizin yazmacını artır.
E00F DF40           STX   40h  ;(5) X dizin yazmacını 40h:41h adresine yükle.
E011 DE42           LDX   42h  ;(4) X dizin yazmacına 42h:43h adresindeki veriyi yükl
E013 A700           STAA  0,X  ;(6) A akümülatörünü (X+0) adresinde sakla.
E015 08             INX        ;(4) X dizin yazmacını artır.
E016 DF42           STX   42h  ;(5) X dizin yazmacını 42h:43h adresine yükle.
E018 8C0039         CPX   #39h ;(3) X dizin yazmacını 39h değeri ile karşılaştır.
E01B 26ED           BNE   L1   ;(4) sıfır değilse L1'e git
E01D 01             NOP        ;(2) sıfır ise programı bitir.
0000                END
```

Programın analizi:

	PC	X	A	Z	(40)	(41)	(42)	(43)	(38)
0	E000	0002	25	1	13	14	15	16	05
1	E005	0030		0	00	30			
2	E00A	0038					00	38	
3	E00F	0030	FC						
4	E011	0031			00	31			
5	E015	0038							FC
6	E01E	0039		1			00	39	

Programın toplam çalışma süresi = (3+5+3+5)+ 1*(4+5+4+5+4+6+4+5+3+4)+2
= 16 + 1*44 + 2 = 62 sistem saati.