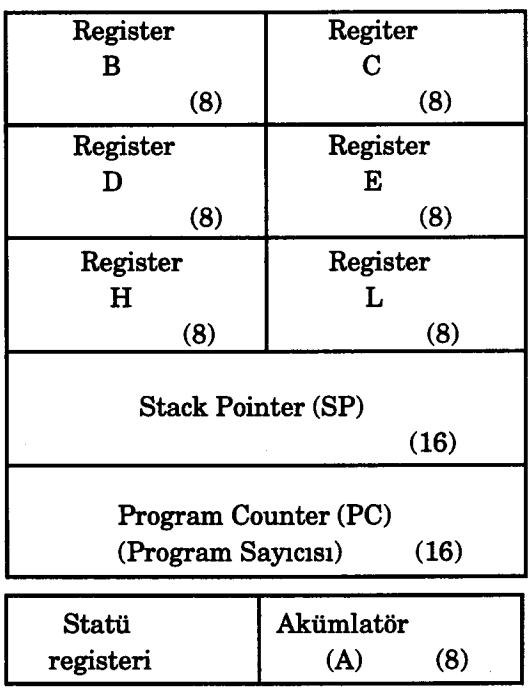
BÖLÜM IV
Intel 8085'in Programlanması ve Komut Listesi

Intel firmasının ürettiği 8 bitlik Mİ'ler içinde en gelişmiş olan 8085'tir. Bu bölümde, Mİ'ye ait komut listesindeki komutlar incelenecek ve yazılım örnekleri verilecektir.

4.1 Yazılım mimarisi:

Yazılım mimarisi, 8085 içinde bulunan ve programlayıcının yazılım içinde kullanılabileceği registerlerden oluşur. Bu mimari Şekil 4.1'de gösterilmiştir.



Şekil 4.1: 8085'in yazılım mimarisi

Şekil 4.1'de 5 adet 16 bitlik register görülmektedir. Bunlardan 3 tanesi genel amaçlıdır ve B,C,D,E,H,L olmak üzere 8 bitlik register olarak kullanılabilirler. Stack Pointer ise RAM içinde oluşturulan yığının adresini saklar. 5'inci 16-bitlik register olan Program Sayıcısı, işlenecek komutun adresini gösterir.

Genel amaçlı 3 adet 16 bitlik register, hem 8 bitlik register olarak, hemde 16 bitlik register olarak, çok sayıda işlemi gerçekleştirirler. Bu registerler 16 bitlik çalışma ile, bellekte erişilecek adresin hesaplanmasında ve 16 bitlik aritmetik işlemlerde kullanılırlar. Aynı registerler 8 bitlik çalışmada, yazılan komuta göre, işleme girecek veriyi saklarlar veya bazı verileri geçici olarak tutarlar. Herhangi bir verinin genel amaçlı registerlerin birisinden alınması, bellekten okunmasından daha az zaman aldığı için, bu registerlerin kullanılması yazılımın hızını artırırır.

Akümlatör: 8085 içinde bazen A register olarak adlandırılan, 8 bitlik bir akümlatör bulunur. 8 bitlik aritmetik, mantık ve kaydırma işlemleri A registeri içinde gerçekleşir. Giriş/Çıkış komutlarında, portlara gönderilen veya portlardan gelen veri akümlatöre yazılır.

Statü registeri: Mİ bir işlemi akümlatör içinde gerçekleştirdiği anda, statü registerine ait bitler, bulunan sonuçla ilgili değerler alırlar. Statü registerinin içeriği Şekil 4.2'de gösterilmiştir. Bu register içindeki her bite **flag** denir.

D7	D6	D5	D4	D3	D2	D1	D0
S	Z		AC		P		CY

Şekil 4.2: Statü registeri

(S) İşaret flağı: Akümlatör içinde gerçekleşen bir aritmetik veya mantık işlemin sonucunda MSB'e bakılır. MSB=1 negatif değeri, MSB=0 pozitif değeri göstermektedir. İşaret flağına 1 yazılması işlemin sonucunun negatif olduğunu ve 0 yazılması sonucun pozitif olduğunu gösterir.

(Z) Sıfır flağı: Akümlatörde bulunan sonucun değeri sıfır ise bu flag 1 değerini alır.

(AC) Yardımcı elde flağı: Bir aritmetik işlem sonucunda 3.

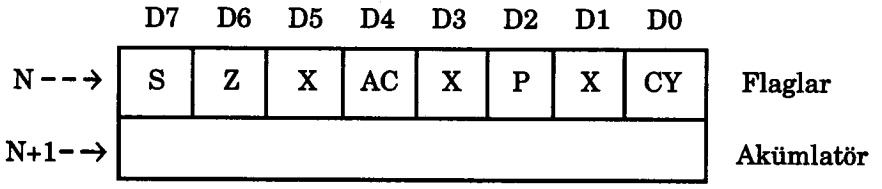
bitten 4. bite bir elde aktarıldığında, yardımcı elde flağına 1 değeri yazılır. Bu flag, BCD (Binary Coded Decimal) sayılar ile çalışırken kullanılır.

(P) Parite flağı: Bir aritmetik veya mantık işleminden sonra bulunan değer içindeki 1 sayısı ile ilgilidir. Eğer sonuç içindeki 1'lerin sayısı çift ise, bu flag 1 değerini alır. Aynı şekilde, bu sayı tek ise parite flağı reset olur. Örneğin, bulunan sonuç 00000011 ise, parite flağı 1 değerini alır. Bu sonucun gerçekte tek sayı olduğuna dikkat ediniz.

(CY) Elde flağı: Bir aritmetik işlemden sonra, MSB'den dolayı oluşan ve akümlatör içine yazılamayan bir elde oluştuğunda, elde flağı 1 değerini alır.

AC (yardımcı elde) dışındaki bütün flaglar yardımı ile, yazılım içinde değişik adreslere atlamak ve değişik işlevleri gerçekleştirmek mümkündür. Yardımcı elde flağı, yanlızca BCD sayılar ile çalışıldığında kullanılır.

Statü registeri ve akümlatörden oluşan 16 bitlik sözcüğe, **Mİ statü sözcüğü (processor status word, PSW)** denir. Mİ statü sözcüğü ve statü registeri farklı ifadelerdir. Gerekli durumlarda Mİ statü sözcüğü rahatlıkla belleğe yazılabilir. Bu sözcüğün, bellekteki görünümü, Şekil 4.3'te gösterilmiştir.



X bitleri tanımlanmamıştır.

Şekil 4.3: Mİ statü sözcüğünün bellekteki görünümü.

4.2 Yığın (stack):

Yığın, Mİ'nin bazı adresleri saklayabilmesi için, RAM içinde ayrılmış özel bir bölümdür. Bu bölüm, programlayıcı tarafından da geçici olarak, verileri saklamak için kullanılabilir. 8085 yığını, bir

interrupt servis programı veya altprogram tamamlandıktan sonra, ana programa yeniden devam edeceği adresi saklamak için kullanır. Örneğin, 8085'e gelen bir interrupt sinyalinden sonra, Mİ adım adım şu işlemleri yapar.

1) Bir sonraki komutu göstermesi için Program Sayıcısını önceden 1-artırmıştır.

2) İşlemekte olduğu komutu tamamlar.

3) Yığının üst sınırını gösteren SP'ı 1- azaltır.

4) PC'ın yüksek mertebeli 8 bitini yığına alır.

5) SP'ı 1-azaltır.

6) PC'ın düşük mertebeli 8 bitini yığına alır.

7) 8085, interrupt servis programına atlar ve bu programı tamamlar.

8) SP'ın gösterdiği adresteki değeri PC'ın düşük mertebeli kısmına yazar.

9) SP, 1-artırılır.

10) SP'ın gösterdiği adresteki değeri PC'ın yüksek mertebeli kısmına alır.

11) SP'ı 1-artırır.

12) Ana programa bıraktığı yerden devam eder.

8085, bir altprograma atlarken yanlızca PC'ı otomatik olarak yığına alır. Programlayıcının kendisinin, Mİ statü sözcüğünü ve diğer registerlerin içeriklerini altprogramın başında, yığına alması ve altprogramın sonunda, aynı içerikleri yığından alıp registerlere tekrar yüklemesi gerekir. Böylece, altprogram süresince statü registerinin ve diğer genel amaçlı registerlerin içeriklerinin değişmesi ana programı etkilemez. Programlayıcının, 8085'in ana programa SP'ın gösterdiği adresin içeriğinden devam edeceğini hatırlayıp, yığına yaptığı giriş sayısı kadar çıkış yapması gerekir. Aksi takdirde, ana programa devam etmek mümkün değildir.

4.4 Komut listesi:

8085 komut, listesi beş gruptan oluşur.

1) **Veri taşıma grubu**: Bu grup içindeki komutlar yardımı ile, bellek ve genel amaçlı registerlerin arasında veri taşınır.

2) **Aritmetik grubu**: Bu komutlar yardımı ile, akümlatörün içeriği ve belirtilen veri arasında aritmetik işlemler yapılır.

3) **Mantık grubu:** Bu grup içindeki komutlar, akümlatörün içeriği üzerinde mantık işlemleri yaparlar.

4) **Satır değiştirme grubu**: Bu komutlar yardımı ile programın akışı içinde değişik satırlara atlanabilir.

5) **Yığın ve Giriş/Çıkış komutları**: Bu grup içindeki komutlar yığına veya çevre elemanına veri göndermek ve oradan veri kabul etmek için kullanılır.

4.4-1 Veri taşıma grubu:

Bu gruba giren komutlar, hem registerler arası hemde, bellek ve register arası veri taşımak için kullanılırlar.

$R_1 \rightarrow R_2$ R_1 ve R_2 genel amaçlı 8 bitlik

Bellek $\rightarrow R_1$ registerlerdir.

$R_1 \rightarrow$ Bellek

Bütün MOV komutlarına ait opcode'larda, en soldaki iki bit 01'dir. Veri taşıma grubu içindeki komutlar, statü registerinin flaglarını etkilemezler.

MOV R_1, R_2 (Move from Register to Register)

Tanım: R_2 registerinin içeriğini R_1'e taşır.

Bellek içinde gerekli register sayısı: 1

Opcode : 01DDDSSS

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 5 periyodu.

Örnek:

MOV A, C (C registerinin içeriği, 44 H)

D (A registeri) 111, S (C registeri) 001

Opcode: 01111001= 79H

Bu komutun işlenmesinden sonra A ve C registerlerinin içerikleri 44 H olacaktır.

MVI R, VERİ (Move Immediate to Register)

Tanım: Komutun ikinci baytında verilen veriyi, R registerine taşır.

Bellek içinde gerekli register sayısı: 2

Opcode: 00DDD110 N N: Komutun bellekte yazılı olduğu adres.

Veri: XXXXXXXX N+1

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 7 periyodu

Örnek:

MVI H, 34 H

D (H registeri) : 100

Opcode: 00100110= 26 H

Veri: 00110100= 34 H

Komutun işlenmesinden sonra, H registerinin içeriği 34 H olacaktır.

MOV R, M (Move to Register from Memory)

Tanım: Bellekten R registerine veri taşır. Taşınacak veriye ait adresin yüksek mertebeli kısmı H, düşük mertebeli kısmı L registerinde bulunur.

Bellek içinde gerekli register sayısı:1

Opcode: 01DDD110

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu

Örnek:

MOV B, M (H registerinin içeriği, 01 H)

(L registerinin içeriği, A5 H)

(Bellek içinde ulaşılacak adres M= 01A5 H)

(Bellekte 01A5 adresindeki veri, 88 H)

D (B registeri)=000

Opcode : 01000110= 46 H

Komutun işlenmesinden sonra, B registerinin içeriği 88 H olacaktır.

MOV M, R (Move to Memory from Register)

Tanım: R registerinin içeriğini belleğe taşır. Bellekte ulaşılacak adresin yüksek mertebeli kısmı H, düşük mertebeli kısmı L registerinde bulunur.

Bellek içinde gerekli register sayısı:1

Opcode: 01110SSS

Adresleme modu: Dolaylı register

İşlenme safhası : Saat sinyalinin 7 periyodu

Örnek:

MOV M, L (H registerinin içeriği 23 H)
(L registerinin içeriği 10 H)
(Bellek içinde erişilecek adres 2310 H)

S (L registeri) : 101

Opcode: 01110101 = 75 H

Komutun işlenmesinden sonra, bellekte 2310 H adresine 10 H değeri yazılmış olacaktır.

MVI M, VERİ (Move Immediate to Memory)

Tanım: Komutun ikinci baytında verilen veriyi belleğe taşır. Bellekte erişilecek adresin yüksek mertebeli kısmı H, düşük mertebeli kısmı L registerinde bulunmalıdır.

Bellek içinde gerekli register sayısı: 2

Opcode: 00110110=36 H N N: Programda komutun
Veri: XXXXXXXX N+1 yazıldığı adres

Adresleme: İçeren ve dolaylı regıster.

İşlenme safhası: Saat sinyalinin 10 periyodu

Örnek:

MVI M, DE H (H registerinin içeriği, 30 H)

(L registerinin içeriği, 00 H)

(Bellekte erişilecek adres, 3000 H)

Opcode: 00110110= 36 H

Veri: 11011110= DE H

Komutun işlenmesinden sonra bellekte 3000 H adresine DE H değeri yazılmış olacak.

LOAD ve STORE komutları da, MOVE komutu gibi registerler arası veri taşıması işlemlerinde kullanılır.

LXI RP, VERİ (Load Immediate to Register Pair)

Tanım: RP; register çiftini göstermektedir. Komutun ikinci baytını register çiftinin düşük mertebeli kısmına, üçüncü baytını ise yüksek mertebeli kısmına yükler.

Bellek içinde gerekli register sayısı: 3

Opcode : 00RP0001 N N: komutun yazıldığı adrestir.

1. Veri : XXXXXXXX N+1

2. Veri : YYYYYYYY N+2

Eğer yüklenen değer bellekteki bir adres ise, 1. veri adresin düşük mertebeli kısmı, 2. Veri adresin yüksek mertebeli kısmıdır.

Adresleme modu: İçeren

İşlenme safhası : Saat sinyalinin 10 periyodu.

Örnek:

LXI D, 1475 H

RP (DE register çifti)= 01

Opcode: 00010001 = 11 H

1. Veri : 01110101 = 75 H

2. Veri : 00010101 = 14 H

Komutun işlenmesinden sonra, D registerinin içeriği 14 H ve E registerinin içeriği 75 H olur.

LDA ADR (Load Accumulator Direct)

Tanım: Bellekteki veriyi akümlatöre taşır. Bellekte erişilecek adresin düşük mertebeli kısmı, komutun ikinci baytında ve yüksek mertebeli kısmı komutun üçüncü baytında verilir.

Bellekte gerekli register sayısı: 3

Opcode: 00111010= 3A H N N: Komutun bellekte yazıl-
1. Veri : XXXXXXXX N+1 dığı adrestir.
2. Veri : YYYYYYYY N+2

1. Veri adresin düşük mertebeli kısmı, 2. Veri adresin yüksek mertebeli kısmıdır.

İşlenme safhası: Saat sinyalinin 13 periyodu.

Örnek:

LDA C003 H (Bellekte erişilecek adres, C003 H)

(Bellekte C003 H adresinin içeriği, 9 A H)

Opcode: 00111010=3A H

1. Veri : 00000011=03 H

2. Veri : 11000000=C0 H

Komutun işlenmesinden sonra, akümlatörün içeriği, 9A H değerini alacaktır.

STA ADR (Store Accumulator Direct)

Tanım: Akümlatörün içeriğini belleğe taşır. Bellekte erişilecek adresin düşük mertebeli kısmı komutun ikinci baytında ve yüksek mertebeli kısmı üçüncü baytında yazılıdır.

Bellek içinde gerekli register sayısı: 3

Opcode: 00110010= 32 H	N,	N: Komutun opcode'unun
1. Veri: XXXXXXXX	N+1	yazıldığı adrestir.
2. Veri: YYYYYYYY	N+2	

1. Veri erişilecek adresin düşük mertebeli kısmı, ve 2. Veri yüksek mertebeli kısmıdır.

Adresleme modu: Doğrudan.

İşlenme safhası: Saat sinyalinin 13 periyodu.

LHLD ADR (Load H and L Direct)

Tanım: Bellekte verilen adrese erişerek, bu adresin içeriğini L registerine alır. Bir sonraki adresin içeriğini de H registerine alır. Erişilecek adresin düşük mertebeli kısmı komutun ikinci baytında ve yüksek mertebeli kısmı üçüncü baytındadır.

Bellek içinde gerekli register sayısı: 3

Opcode: 00101010=2AH	N	N: komutun opcoda'unun
1. Veri: XXXXXXXX	N+1	yazıldığı adrestir.
2. Veri: YYYYYYYY	N+2	

1. Veri erişilecek adresin düşük mertebeli kısmı ve 2. Veri yüksek mertebeli kısmıdır.

Adresleme modu: Doğrudan

İşlenme safhası: Saat sinyalinin 16 periyodu.

Örnek:

LHLD C6A7 H (Erişilecek bellek adresi, C6A7H)
(C6A7H adresinin içeriği 00H)
(Erişilecek bir sonraki adres, C6A8H)
(C6A8H adresinin içeriği, 27 H)

Opcode: 00101010= 2A H

1. Veri : 10100111= A7 H

2. Veri : 11000110= C6 H

Komutun işlenmesinden sonra H registerinin içeriği, 27H ve L registerinin içeriği 00H olacaktır.

SHLD ADR (Store H and L Direct)

Tanım: İlk önce verilen bellek adresine erişerek, L registerinin içeriğini bu adrese taşır. Bundan bir sonraki adrese de H registerinin içeriğini aktarır. Erişilecek adresin düşük mertebeli kısmı komutun ikinci baytında ve yüksek mertebeli kısmı üçüncü baytında bulunur.

Bellek içinde gerekli register sayısı: 3

Opcode: 00100010=22 H	N,	N: Komutun opcode'unun
1. Veri : XXXXXXXX	N+1	yazıldığı adrestir.
2. Veri : YYYYYYYY	N+ 2	

1. Veri erişilecek adresin düşük mertebeli kısmını, 2. Veri yüksek mertebeli kısmını gösterir.

Adresleme modu: Doğrudan

İşlenme safhası: Saat sinyalinin 16 periyodu.

LDAX RP (Load Accumulator Indirect)

Tanım: Bellek adresi RP register çiftinde verilen veriyi bellekten akümlatöre alır. (Yanlız BC veya DE, RP olarak kullanılabilir.)

Bellek içinde gerekli register sayısı: 1

Opcode: 00RP1010

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu

Örnek:

LDAX B (B registerinin içeriği, 78 H)
(C registerinin içeriği, 12 H)
(Erişilecek bellek adresi, 7812 H)
(Bellekte 7812 H adresinin içeriği, B6 H)

RP (BC register çifti) : 00H

Opcode: 00001010= 0A H

Komutun işlenmesinden sonra akümlatörün içeriği B6 H olacaktır.

STAX RP (Store Accumulator Indirect)

Tanım: Akümlatörün içeriğini belleğe taşır. Erişilecek adres, RP register çiftinde yazılıdır.

Bellek içinde gerekli register sayısı: 1

Opcode : 00RP0010

Adresleme modu : Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu

XCHG (Exchange H and L with D and E)

Tanım: H registerinin içeriği D registerine, D registerinin içeriği H registerine alınır. Aynı işlem L ve E registerleri arasında da gerçekleşir.

Bellek içinde gerekli register sayısı: 1

Opcode: 11101011= EB H

Adresleme modu: Register

İşlenme safhası : Saat sinyalinin 4 periyodu.

Örnek:

XCHG (H registerinin içeriği 23 H)
(L registerinin içeriği 67 H)
(D registerinin içeriği 9A H)
(E registerinin içeriği F4 H)

Opcode: 11101011= EB H

Komutun işlenmesinden sonra H registerinin içeriği 9A H, L registerinin içeriği F4H, D registerinin içeriği 23 H ve E registerinin içeriği 67 H olacaktır.

4.4-2 Aritmetik işlemler grubu:

Bu grup içindeki komutlar; toplama, çıkarma kaydırma vb. işlemleri yaparlar. Bu işlemlerden etkilenen flaglar ilgili komutlarda verilmiştir. Çıkarma işlemlerinde ikinin tümleyeni aritmetiği kullanılır.

ADD R (Add Register)

Tanım: R registerinin içeriği ile akümlatörün içeriği toplanır ve sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10000SSS

Etkilenen flaglar: Sıfır, işaret, parite, elde, yardımcı elde.

Adresleme modu: Register

İşlenme safhası : Saat sinyalinin 4 periyodu

Örnek:

ADD C

S(C registeri) = 001

Opcode: 10000001= 81 H

A reg. : 10001001= 89 H

C reg. : 11000011= C3 H

Komutun işlenmesinden sonra, akümlatörün içeriği 01001100=4C H olacaktır. Elde flağı 1, diğer flaglar 0 değerini alırlar.

ADD M (Add Memory)

Tanım: Bellekten, adresi HL register çiftinde bulunan değeri okuyup, akümlatörün içeriği ile toplar. Sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10000110=86 H

Etkilenen flaglar: Sıfır, işaret, parite, elde ve yardımcı elde

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu.

ADI VERİ (Add Immediate)

Tanım: Akümlatörün içeriği ile komutun ikinci baytında verilen veri toplanıp, sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 2

Opcode : 11000110= C6 H

Veri : XXXXXXXX

Etkilenen flaglar: Sıfır, işaret, parite, elde, yardımcı elde

Adresleme mode: İçeren

İşlenme safhası : Saat sinyalinin 7 periyodu.

Örnek:

ADI 44 H

Opcode: 11000110= C6 H

Veri: 01000100= 44 H

A reg: 00010001= 11 H

Komutun işlenmesinden sonra akümlatörün içeriği 01010101= 55 H olacaktır. Parite flağı 1, diğer flaglar 0 değerini alırlar.

ADC R (Add Register with Carry)

Tanımı: R registerinin içeriğini ve elde flağının değerini akümlatörün içeriği ile toplar ve sonucu akümlatörde saklar.

Bellek içinde gerekli register sayısı: 1

Opcode: 10001SSS

Etkilenen flaglar: Sıfır, işaret, parite, elde ve yardımcı elde

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 4 periyodu.

Örnek:

ADC E

S (E registeri)= 011

Opcode : 10001011= 8B H

A reg. içeriği: 00011100= 1C H

E reg. içeriği: 00110000= 30 H

Elde flağı : 1= 01 H

Komutun işlenmesinden sonra A reg. içeriği=01001101= 4D H

Parite flağı 1, diğer flaglar 0 değerini alır.

ADC M (Add Memory with Carry)

Tanım: Bellekten adresi HL register çiftinde verilen değeri okuyup, akümlatörün içeriği ve elde flağının değeri ile toplar. Sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10001110 = 8E H

Etkilenen flaglar: Sıfır, işaret, parite, elde ve yardımcı elde

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu.

ACI VERİ (Add Immediate with Carry)

Tanım: Komutun ikinci baytında verilen veri ile elde flağının değeri ve akümlatörün içeriği toplanır. Sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 2

Opcode: 11001110= CE H

Etkilenen flaglar: Sıfır, işaret, parite, elde ve yardımcı elde

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 7 periyodu

SUB R (Subtract Register)

Tanım: R registerinin içeriğini akümlatörün içeriğinden çıkarır ve sonuç akümlatörde saklanır. R registeri genel amaçlı 8 bitlik registerlerden birisidir.

Bellek içinde gerekli register sayısı: 1

Opcode: 10010SSS

Etkilenen flaglar: Sıfır, işaret, parite, elde, yardımcı elde

Adresleme modu: Register

İşlenme safhası : Saat sinyalinin 4 periyodu.

Örnek:

SUB D

S (D registeri): 010

Opcode : 10010010= 92 H

A reg. içeriği: 10000111= 87 H

D reg. içeriği: 10101011= AB H

Komutun işlenmesinden sonra A reg. içeriği= 11011100= DCH olacaktır. İşaret ve elde flagları 1, diğer flaglar 0 değerini alırlar.

SUB M (Subtract Memory)

Tanım: Adresi HL register çiftinde verilen değeri bellekten okur ve bu değeri akümlatörün içeriğinden çıkarır. Sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10010110= 96 H

Etkilenen flaglar: Sıfır, işaret, parite, elde ve yardımcı elde.

Adresleme mode: Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu.

SUI VERİ (Subtract Immediate)

Tanım: Komutun ikinci baytında verilen veriyi, akümlatörün içeriğinden çıkarır ve sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 2

Opcode: 11010110= D6 H

Etkilenen flaglar: Sıfır, işaret, parite, elde ve yardımcı elde.

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 7 periyodu.

SBB R (Subtract Register with Borrow)

Tanım: R registerinin içeriğini ve elde flağının değerini akümlatörün içeriğinden çıkarır ve sonucu akümlatörde saklar.

Bellek içinde gerekli register sayısı: 1

Opcode: 10011SSS

Etkilenen flaglar: Sıfır, işaret, parite, elde ve yardımcı elde.

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 4 periyodu.

Örnek:

SUB B

S (B registeri): 000

Opcode : 10011000= 98 H

A reg. içeriği: 00010001= 11 H

B reg. içeriği: 00000101= 05 H

Elde flağı : 1= 01 H

Komutun işlenmesinde sonra akümlatörün içeriği 00001011= 0B H olur. Bütün flaglar 0 değerini alır.

SBB M (Subtract Memory with Borrow)

Tanım: Adresi HL register çiftinde verilen değeri bellekten okur ve bu değer ile birlikte elde flağının değerini akümlatörün içeriğinden çıkarır. Sonuç, akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10011110=9EH

Etkilenen flaglar: Sıfır, işaret, parite, elde ve yardımcı elde.

Adresleme modu: Dolaylı register.

İşlenme safhası: Saat sinyalinin 7 periyodu

SBI VERİ (Subtract Immediate with Borrow)

Tanım: Komutun ikinci baytında verilen veri ile birlikte elde flağının değerini akümlatörün içeriğinden çıkarır. Sonucu akümlatörde saklar.

Bellek içinde gerekli register sayısı: 2

Opcode: 11011110= DE H

Etkilenen flaglar: Sıfır, işaret, parite, elde ve yardımcı elde

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 7 periyodu.

Örnek:

SBI 00H

Opcode : 11011110= DE H

Veri: 00000000= 00 H

A reg. : 01010001= 51 H

Elde flağı : 1= 01 H

Komutun işlenmesinden sonra akümlatörün içeriği 01010000= 50 H olur. Parite flağı 1, diğer flaglar 0 değerini alırlar.

DAD RP (Add Register Pair to H and L)

Tanım: RP register çiftinin 16 bitlik içeriğini, HL register çiftinin içeriği ile toplar ve sonucu HL register çiftinde saklar.

Bellek içinde gerekli register sayısı: 1

Opcode: 00RP1001

Etkilenen flag: Elde

Adresleme modu: Register

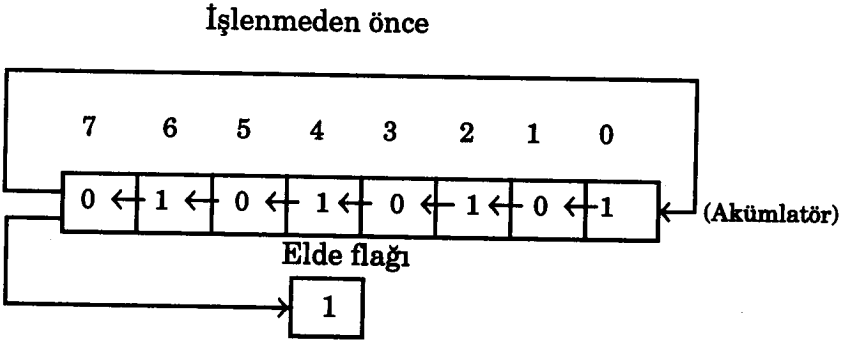
İşlenme safhası: Saat sinyalinin 10 periyodu.

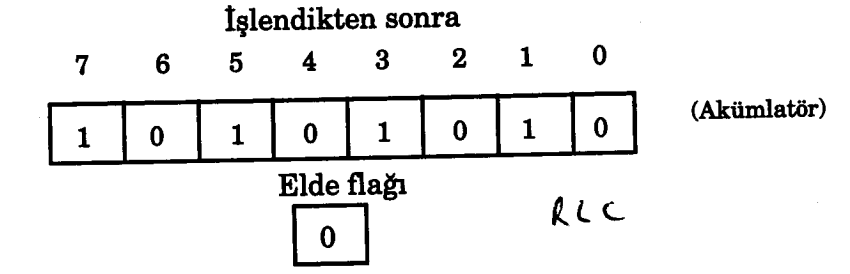
Örnek: DAD RP komutu Stack Pointerin içeriğini saklamak için kullanılabilir.

```
LXI H 00H      : HL register çiftinin içeriğini 0 yap.
DAD  SP        : Stack Pointerin içeriğini HL'e al.
SHLD 3E00H     : Stack Pointerin içeriğini bellekte
                 3E00H adresinde sakla.
```

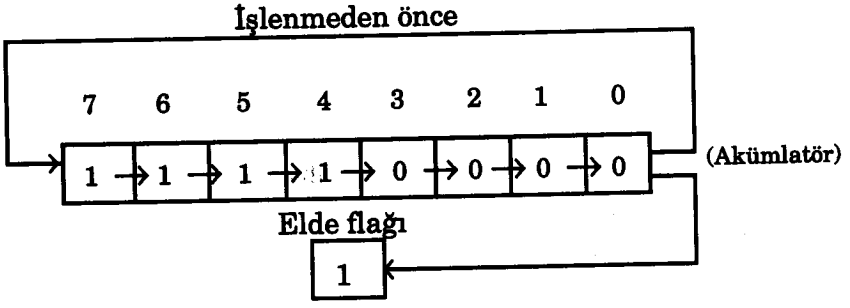
Komutun işlenmesinden sonra Stack Pointerin içeriği bellekte 3E00H ve 3E01H adreslerinde saklanır. Bu değer, aynı zamanda, HL register çiftinde de bulunur. DAD H komutu MSB'den elde gelmediği durumda HL register çiftinin içeriğini 2 ile çarpmış olmak için kullanılabilir.

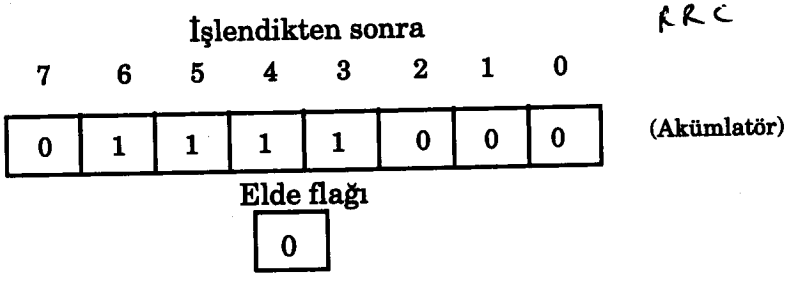
Kaydırma komutları: 8085 komut listesinde dört adet kaydırma (ROTATE) komutu vardır. Bunlar Şekil 4.6'da gösterilmiştir.



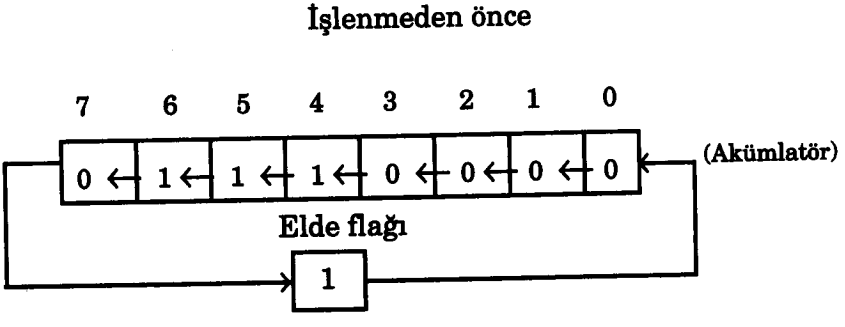

İşlendikten sonra
7 6 5 4 3 2 1 0
1 0 1 0 1 0 1 0
(Akümlatör)
Elde flağı
0
RLC

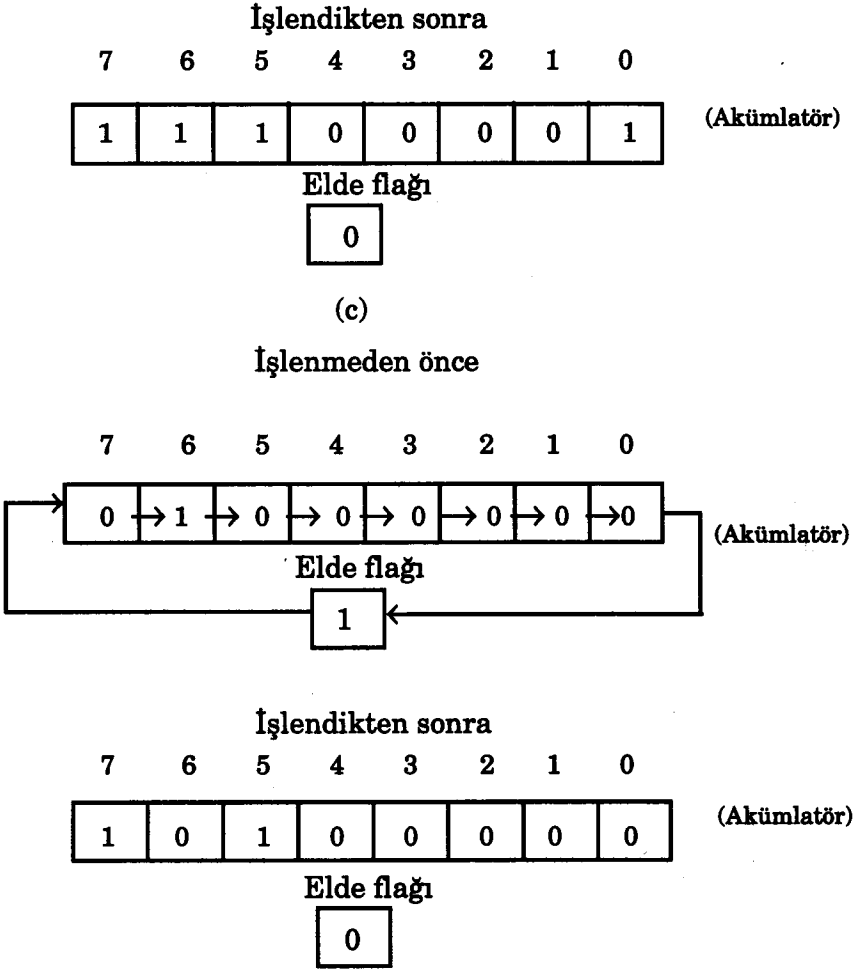
(a)


İşlenmeden önce
7 6 5 4 3 2 1 0
1 1 1 1 0 0 0 0
(Akümlatör)
Elde flağı
1


RRC
İşlendikten sonra
7 6 5 4 3 2 1 0
0 1 1 1 1 0 0 0
(Akümlatör)
Elde flağı
0

(b)


İşlenmeden önce
7 6 5 4 3 2 1 0
0 1 1 1 0 0 0 0
(Akümlatör)
Elde flağı
1



Şekil 4.6: Kaydırma komutlarının gösterimi a) RLC komutu, b) RRC komutu, c) RAL Komutu, d) RAR komutu

RLC (Rotate Accumulator Left)

Tanım: Akümlatörün içeriğini bir bit sola döndürerek, MSB'in değerini LSB'e taşır. Elde flağı, MSB'in kaydırmadan önceki değerini alır.

Bellek içinde gerekli register sayısı: 1

Opcode: 00000111= 07 H

Etkilenen flag: Elde

İşlenme safhası : Saat sinyalinin 4 periyodu

Örnek: RLC

A reg. 01010101= 55 H

Elde flağı= 1

Komutun işlenmesinden sonra

A reg.= 10101010= AA H

Elde flağı= 0

RRC (Rotate Accumulator Right)

Tanım: İşlem RLC komutu ile benzer şekildedir. Bu komutta kaydırma işlemi sağa doğru, LSB'in değeri MSB ve elde flağına taşınacak şekilde gerçekleşir.

Opcode: 00001111= 0F H

Etkilenen flag: Elde flağı

İşlenme safhası: Saat sinyalinin 4 periyodu

RAL (Rotate Left through Carry)

Tanım: Akümlatörün içeriğini elde flağı ile birlikte 1 bit sola kaydırır. Elde flağı akümlatörün bir parçası gibi işlenir. Akümlatördeki MSB, elde flağına taşınırken, bu flağın eski değeri LSB'e alınır.

Opcode: 00010111= 17 H

Etkilenen flaglar: Elde flağı

İşlenme safhası: Saat sinyalinin 4 periyodu.

RAR (Rotate Right through Carry)

Tanım: İşlem RAL komutuna benzer şekildedir. Akümlatö-

rün içeriği, LSB elde flağına, elde flağı da MSB'e alınacak şekilde sağa kaydırılır.

Opcode: 00011111: 1F H

Etkilenen flaglar: Elde flağı

İşlenme safhası: Saat sinyalinin 4 periyodu

8085 komut listesinde 1-artırma ve 1-azaltma komutları vardır.

INR R (Increment Register)

Tanım: INR R, belirli bir registerin içeriğini 1 ile toplar.

Opcode: 00DDD100

Etkilenen flaglar: Elde flağı dışındaki tüm flaglar.

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 5 periyodu.

Örnek: INR B

Opcode: 00000100=04 H

Eğer B registerinin içeriği 39 H ise, komutun işlenmesinden sonra bu değer 3A H olacaktır.

INR M (Increment Memory)

Tanım: HL register çiftinin, bellekte gösterdiği adresin içeriği, 1 ile toplanır, (1-artırılır).

Bellek içinde gerekli register sayısı: 1

Opcode: 00110100= 34 H

Etkilenen flaglar: Elde flağı dışında tüm flaglar

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 10 periyodu

Örnek:

INR M

Eğer HL register çiftinin içeriği 3E10H ise, işlemden sonra bellekte 3E10H adresinin içeriği 1 artırılacaktır.

INX RP (Increment Register Pair)

Tanım: Belirlenmiş 16 bitlik bir register çiftinin içeriğini 1 ile toplar.

Bellek içinde gerekli register sayısı: 1

Opcode: 00RP0011

Etkilenen flaglar: Hiçbirisi

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 5 periyodu

Örnek:

INX B

Opcode: 00000011= 03 H

Eğer BC register çiftinin içeriği 1FFFH ise, işlemden sonra BC register çiftinin içeriği 2000 H olacaktır.

Not: INX ve DEX komutlarının flagları etkilememesi, register çiflerinin sayıcı olarak kullanılmaları sırasında gözönüne alınmalıdır.

DCR R (Decrement Register)

Tanım: Belirlenen registerin içeriğinden 1 çıkarır.

Bellek içinde gerekli register sayısı= 1

Opcode: 00DDD101

Etkilenen flaglar: Elde flağı dışında, tüm flaglar etkilenir.

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 5 periyodu

DCR M (Decrement Memory)

Tanım: HL register çiftinin bellekte gösterdiği adresin içeriğinden 1 çıkarır.

Bellek içinde gerekli register sayısı: 1

Opcode: 00110101= 35 H

Etkilenen flaglar: Elde flağı dışında tüm flaglar

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 10 periyodu

DCX RP (Decrement Register Pair)

Tanım: Belirli bir register çiftinin 16 bitlik içeriğini 1-azaltır.

Bellek içinde gerekli register sayısı: 1

Opcode: 00RP1011

Etkilenen flaglar: Hiçbirisi

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 5 periyodu.

CMA (Complement Accumulator)

Tanım: Akümlatördeki sayının tümleyenini alır.

Bellek içinde gerekli register sayısı: 1

Opcode: 00101111= 2F H

Etkilenen flaglar: Hiçbirisi

İşlenme safhası: Saat sinyalinin 4 periyodu

Örnek:

CMA

A reg.= 11000111= C7 H

Komutun işlenmesinden sonra;

A reg= 00111000= 38 H

STC (Set Carry)

Tanım: Elde flağının değerini 1 yapar.

Bellek içinde gerekli register sayısı: 1

Opcode: 00110111= 37 H

Etkilenen flaglar: Elde flağı

İşlenme safhası: Saat sinyalinin 4 periyodu

CMC (Complement Carry)

Tanım: Eğer elde flağının değeri 0 ise 1, 1 ise 0 yapar.

Bellek içinde gerekli register sayısı: 1

Opcode: 00111111=3F H

Etkilenen flaglar: Elde flağı

İşlenme safhası: Saat sinyalinin 4 periyodu

DAA (Decimal Adjust Accumulator)

Tanım: 8 bitlik akümlatörün içeriğini, BCD sayılar ile çalışacak şekilde düzenler.

Bellek içinde gerekli register sayısı= 1

Opcode: 00100111= 27 H

Etkilenen flaglar: Bütün flaglar

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 4 periyodu

DAA'ın çalışması şöyledir;

1) Yardımcı elde flağının değeri 1 ise veya akümlatörün içeriğinde düşük mertebeli 4 bitlik bölümün değeri 9'dan büyük ise; akümlatörün içeriğine 6 eklenir.

2) Elde flağının değeri 1 ise veya akümlatörün içeriğinde yüksek mertebeli 4 bitlik kısmın değeri 9'dan büyük ise, bu kısma 6

eklenir.

Örnek:

8085 içinde, BCD olarak verilmiş 48 ve 59 sayıları nasıl toplanır?

Çözüm: İlk önce sayılar ikili sistemde olduğu gibi toplanırlar.

```
  48
+ 59
------
  A1 H
```

Bu toplama sonucunda elde flağı 0 ve yardımcı elde flağı 1 değerini alır. Dolayısıyla, her iki dörtlü gruba da 6 eklenmesi gerekir.

```
  A1
+ 66
------
  07
```

Bu toplama elde flağını etkiler ve sonuç beklendiği gibi 107 olarak bulunur.

4.4-3 Lojik grubu:

8085 mikroişlemcisinde kullanılabilen lojik komutları AND, OR ve XOR dur.

Not: Bu grupta bulunan komutlar, bütün flagları etkilerler.

ANA R (AND Register)

Tanım: ANA komutu, belirli bir registerin içeriği ile akümlatörün içeriği arasında VE işlemi yapar. Sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10100SSS

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 4 periyodu

ANA M (AND Memory)

Tanım: HL register çiftinde verilmiş olan adresin içeriği ile akümlatörün içeriği arasında VE işlemi yapar. Sonuç akümlatörde kalır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10100110= A6 H

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu

ANI VERİ (AND Immediate)

Tanım: Komutun ikinci baytı ile akümlatör arasında, VE işlemi yapar. Sonuç akümlatörde kalır.

Bellek içinde gerekli register sayısı: 2

Opcode: 11100110= E6 H

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 7 periyodu

XRA R (EXCLUSIVE-OR Register)

Tanım: Belirli bir registerin içeriği ile akümlatörün içeriği arasında EXCLUSIVE-OR işlemi yapar. Sonuç akümlatörde kalır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10101SSS

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 4 periyodu

XRA M (EXCLUSIVE-OR Memory)

Tanım: HL register çiftinde verilmiş olan adresin içeriği ile akümlatörün içeriği arasında EXCLUSIVE-OR işlemi yapar. Sonuç akümlatörde kalır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10101110= AE H

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu

XRI VERİ (EXCLUSIVE-OR Immediate)

Tanım: Komutun ikinci baytı ile akümlatörün içeriği arasında EXCLUSIVE-OR işlemi yapar. Sonuç akümlatörde kalır.

Bellek içinde gerekli register sayısı: 2

Opcode: 11101110=EE H

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 7 periyodu

ORA R (OR Register)

Tanım: belirli bir registerin içeriği ile akümlatörün içeriği arasında VEYA işlemi yapar. Sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10110SSS

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 4 periyodu

ORA M (OR Memory)

Tanım: HL register çiftinde yazılı olan adresin içeriği ile akümlatör arasında VEYA işlemi yapar. Sonuç akümlatörde kalır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10110110= B6 H

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu

ORI VERİ (OR Immediate)

Tanım: Komutun ikinci baytı ile akümlatörün içeriği arasında VEYA işlemi yapar. Sonuç akümlatörde saklanır.

Bellek içinde gerekli register sayısı: 2

Opcode: 11110110= F6 H

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 7 periyodu.

Karşılaştırma (Compare) komutları ile akümlatörün içeriği, bir registerin içeriği veya bir veri ile karşılaştırılır. Bu işlem sonucunda registerin ve akümlatörün içerikleri değişmez. Bununla birlikte flaglar sonucu belirtir.

CMP R (Compare Register)

Tanım: Belirli bir registerin içeriği ile akümlatörün içeriğini karşılaştırır. Sonuç flagların durumu ile gösterilir. Flagların durumları aşağıda verilmiştir.

Flag	Değeri
Sıfır (Z)	1: A reg. içeriği= R içeriği
	0: A reg. içeriği≠R içeriği
Elde (CY)	1: A reg. içeriği > R içeriği
	0: A reg. içeriği< R içeriği

Eğer karşılaştırılan sayıların işaretleri farklı ise, elde flağı aynı değerleri yukarıda verilen koşulların tam tersinde alır.

Bellek içinde gerekli register sayısı: 1

Opcode: 10111SSS

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 4 periyodu

CMP M (Compare Memory)

Tanım: HL register çiftinde verilmiş olan adresin içeriği ile

akümlatörü karşılaştırır. Sonuç, flagları CMP R komutunda olduğu gibi etkiler.

Bellek içinde gerekli register sayısı: 1

Opcode: 10111110= BEH

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 7 periyodu

CPI VERİ (Compare Immediate)

Tanım: Komutun ikinci baytı ile akümlatörü karşılaştır. Sonuç flagları CMP R komutunda olduğu gibi etkiler.

Bellek içinde gerekli register sayısı: 2

Opcode: 11111110= FEH

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 7 periyodu

4.4-4 Satır değiştirme komutları:

Bu grup içerisinde JUMP, CALL, RETURN ve RESTART komutları vardır. Bu komutlardan hiçbirisi flagları etkilemez. Komut ile birlikte bir adres belirtildiği durumda, adresin düşük mertebeli kısmı komutun ikinci baytını ve adresin yüksek mertebeli kısmı komutun üçüncü baytını oluşturur.

JMP ADR (JUMP)

Tanım: Komutun ikinci ve üçüncü baytında verilmiş adres, Program Sayıcısına aktarılır. Mİ programa bu adresten devam eder.

Bellek içinde gerekli register sayısı: 3

Opcode: 11000011= C3 H

$ADRES_1$= Düşük mertebeli kısım

$ADRES_2$ =Yüksek mertebeli kısım

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 10 periyodu

J(FLAG) ADR (Jump Conditional)

Tanım: Eğer (FLAG) ifadesi ile gösterilen koşul gerçekleşmiş ise, Program Sayıcısına, komutun opcode'unun arkasından yazılmış iki baytlık adres yüklenir. Eğer bu koşul gerçekleşmemiş ise, yazılıma sıradaki komut işlenerek devam edilir. Bu grupta sıfır, elde, parite ve işaret flaglarının değerine bağlı olan sekiz adet komut bulunur. Bu koşullu satır değiştirme komutları, aşağıda verilmiştir.

Komut	Opcode	Koşul	Hex
JNZ	11000010	Sıfır değil ise atla (Z=0)	C2 H
JZ	11001010	Sıfır ise atla (Z=1)	CA H
JNC	11010010	Elde yok ise atla (CY=0)	D2 H
JC	11011010	Elde var ise atla (CY=1)	DA H
JPO	11100010	Tek parite ise atla (P=0)	E2 H
JPE	11101010	Çift parite ise atla (P=1)	EA H
JP	11110010	Pozitif ise atla (S=0)	F2 H
JM	11111010	Negatif ise atla (S=1)	FA H

Bellek içinde gerekli register sayısı: 3 (Opcode+2 adr. baytı)

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 10 periyodu

PCHL (Move H and L to Program Counter)

Tanım: HL register çiftinin içeriğini Program Sayıcısına alır ve Mİ programa bu adresten devam eder.

Bellek içinde gerekli register sayısı: 1

Opcode: 11101001= E9 H

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 5 periyodu

CALL ADR (Call Subroutine Immediate)

Tanım: Sıradaki komutun adresini yığına aldıktan sonra, CALL komutunun ikinci ve üçüncü baytlarında belirtilen adresdeki komuttan programa devam eder.

Bellek içinde gerekli register sayısı: 3

Opcode: 11001101= CDH

$ADRES_1$= Düşük mertebeli kısım

$ADRES_2$= Yüksek mertebeli kısım

Adresleme modu: İçeren/dolaylı register

İşlenme safhası: Saat sinyalinin 17 periyodu

C(FLAG) ADR (Call Subroutine Conditional)

Tanım: Eğer komut ile birlikte verilen koşul sağlanmış ise, sıradaki komutun adresi yığına alındıktan sonra, CALL komutunun ikinci ve üçüncü baytlarında belirtilen adres, Program Sayıcısına yüklenir. Bu koşullar, koşullu JUMP komutlarındaki sekiz koşul ile aynıdır. Koşullu CALL komutları aşağıda verilmiştir.

Komut	Opcode	Hex
CNZ	11000100	C4 H
CZ	11001100	CC H
CNC	11010100	D4 H
CC	11011100	DC H
CPO	11100100	E4 H
CPE	11101100	EC H
CP	11110100	F4 H
CM	11111100	FC H

Bellek içinde gerekli register sayısı:3 (Opcode+2 adr. baytı)

Adresleme modu: İçeren/dolaylı register

İşlenme safhası: Saat sinyalinin 11 periyodu (Koşul sağlanmamış ise)

Saat sinyalinin 17 periyodu (Koşul sağlanmış ise)

RET (Return Immediate)

Tanım: Yığının en üstteki baytı Program Sayıcısına alınır ve programa yeni adresten devam edilir. Bu komut, herhangi bir CALL komutu veya bir interrupt sinyali ile bir altprograma geçtikten sonra, altprogramın sonunda kullanılır.

Bellek içinde gerekli register sayısı:1

Opcode: 11001001= C9 H

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 10 periyodu.

R (FLAG) (Return Conditional)

Tanım: Eğer komut ile verilen koşul gerçekleşmiş ise, RET komutu işlenir. Diğer durumda sıradaki komut işlenir. Koşullu JMP komutu için verilen koşullar, bu komut için de geçerlidir.

Komut	Opcode	Hex
RNZ	11000000	C0 H
RZ	11001000	C8 H
RNC	11010000	D0 H
RC	11011000	D8 H
RPO	11100000	E0 H
RPE	11101000	E8 H
RP	11110000	F0 H
RM	11111000	F8 H

Bellek içinde gerekli register sayısı: 1

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 5 periyodu (Koşul sağlanmamış ise).
Saat sinyalinin 11 periyodu (Koşul sağlanmış ise).

RST n (Restart)

Tanım: RST, Program Sayıcısının içeriğini yığına aktardıktan sonra, programda aşağıda verilen sekiz adresten birine atlıyan özel amaçlı bir CALL komutudur. Bu komut ile atlanacak adres, komutun opcode'undaki 3., 4. ve 5. bitler ile belirtilir. RST komutları ve bu komutlar ile atlanacak adresler aşağıda verilmişlerdir.

Komut	Opcode	Atlanacak adres
RST 0	11000111	0000 H
RST 1	11001111	0008 H
RST 2	11010111	0010 H
RST 3	11011111	0018 H
RST 4	11100111	0020 H
RST 5	11101111	0028 H
RST 6	11110111	0030 H
RST 7	11111111	0038 H

Örnek:

RST 4 komutunun 8085 Mİ'si için yazılmış bir programdaki etkisini belirtiniz.

Çözüm:

Bu komut işlenirken, SP'ın içeriği iki defa azaltılarak, Program Sayıcının içeriği yığına alınır ve 0020 H değeri Program Sayıcısına yüklenir.

RST komutları, 8085 Mİ'sinin donanımının anlatıldığı Bölüm V'de, daha ayrıntılı olarak incelenecektir.

4.4-5 Yığın, Giriş/Çıkış ve makina kontrolü

Bu komutlar yığını, SP'ı ve Giriş/Çıkış işlemlerini kontrol eden komutlardır.

PUSH RP (Push Register Pair)

Tanım: SP'ın içeriğini 1-azalttıktan sonra, komut ile belirtilen register çiftinin yüksek mertebeli kısmını bellekte SP'ın gösterdiği adrese yazar. Daha sonra SP'ın içeriğini birkez daha 1-azaltır ve bu kez gösterdiği adrese de register çiftinin düşük mertebeli kısmını yazar. Eğer komutun opcode'unda RP olarak belirtilen bitlere 11 yazılırsa, PSW yığına aktarılır.

Bellek içinde gerekli register sayısı: 1

Opcode: 11RP0101

Etkilenen flaglar: Hiçbirisi

Adresleme modu: Dolaylı register

Komutun işlenme safhası: Saat sinyalinin 11 periyodu

POP RP (Pop Register Pair)

Tanım: PUSH komutunun gerçekleştirdiği işlemin tam tersini gerçekleştirir. Düşük mertebeli kısım yığından alınır ve belirtilen registere aktarılır. Daha sonra SP'ın içeriği 1-artırılır ve yüsek mertebeli kısım yığından alınarak, ilgili registere yazılır. Bundan sonra SP'ın içeriği bir kez daha 1-artırılır. Eğer komutun opcode'unda RP olarak belirtilen bitlere 11 yazılırsa PSW, akümlatör ve flag registerine aktarılır.

Bellek içinde gerekli register sayısı: 1

Opcode: 11RP0001

Etkilenen flaglar: Hiçbirisi

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 10 periyodu

XTHL (Exchange H and L with Top of Stack)

Tanım: Yığının en üstteki iki baytı ile HL register çiftinin içeriği yer değiştirir.

Bellek içinde gerekli register sayısı: 1

Opcode: 11100011= E3 H

Etkilenen flaglar: Hiçbirisi

Adresleme modu: Dolaylı register

İşlenme safhası: Saat sinyalinin 18 periyodu

SPHL (Move H and L to Stack Pointer)

Tanım: HL register çiftinin içeriğini SP'a aktarır.

Bellek içinde gerekli register sayısı: 1

Opcode: 11111001= F9 H

Etkilenen flaglar: Hiçbirisi

Adresleme modu: Register

İşlenme safhası: Saat sinyalinin 5 periyodu

OUT PORT (Output to Port)

Tanım: Akümlatörün içeriğini, adresi komutun ikinci baytında verilen porta gönderir.

Bellek içinde gerekli register sayısı: 2

Opcode: 11010011= D3 H

PORT ADRESİ (8 bit)

Etkilenen flaglar: Hiçbirisi

Adresleme modu: Doğrudan

İşlenme safhası: Saat sinyalinin 10 periyodu

IN PORT (Input from Port)

Tanım: Adresi komutun ikinci baytına verilen port içindeki değeri akümlatöre alır.

Bellek içinde gerekli regiter sayısı: 2

Opcode: 11011011= DB H

PORT ADRESİ (8 bit)

Etkilenen flaglar: Hiçbirisi

Adresleme modu: Doğrudan

İşlenme safhası: Saat sinyalinin 10 periyodu

DI (Disable Interrupts)

Tanım: 8085 Mİ'si içinde, bazı durumlarda interrupt sinyallerini maskelemek için bir flıp-flop bulunur. DI komutunun işlenmesi ile bu flip-flop OFF duruma geçer ve 8085 dışarıdan gelen maskelenebilir interrupt sinyallerine tepki göstermez.

Bellek içinde gerekli register sayısı: 1

Opcode: 11110011= F3 H

Etkilenen flaglar: Hiçbirisi

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 4 periyodu

EI (Enable Interrupts)

Tanım: Bu komut, interrup flip-flobunu ON duruma getirerek, Mİ'nin dışarıdan gelen interrupt sinyallerini kabul etmesini sağlar. Bu komut interrupt servis altprogramının sonunda kullanılabilir.

Bellek içinde gerekli register sayısı: 1

Opcode: 11111011= FB H

Etkilenen flaglar: Hiçbirisi

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 4 periyodu

NOP (No Operation)

Tanım: Bu komut hiçbir işlem yapmaz. Zamanlama programlarında Mİ'yi bekletmek için kullanılır.

Bellek içinde gerekli register sayısı: 1

Opcode: 00000000= 00 H

Etkilenen flaglar: Hiçbirisi

İşlenme safhası: Saat sinyalinin 4 periyodu

HLT (Halt)

Tanım: Bu komutun işlenmesinden sonra, Mİ çalışmasını durdurur. Program Sayıcısı bir sonraki komutun adresini göstermektedir. Eğer interrupt girişleri maskelenmemiş ise, dışarıdan gelen bir interrupt sinyali, Mİ'nin yeniden çalışmasını sağlar. Bu girişlerin maskelendiği durumda, Mİ sadece Reset sinyalinden sonra yeniden çalışmaya başlar.

Bellek içinde gerekli register sayısı: 1

Opcode: 01110110= 76 H

Adresleme modu: İçeren

İşlenme safhası: Saat sinyalinin 7 periyodu

Buraya kadar, 8085'in programlanması ile ilgili komutlar verildi. Bölüm V'de, bu Mİ'in donanım yapısı açıklanırken, donanım ile ilgili birkaç komut daha bu listeye eklenecektir. Aşağıdaki bölümde 8085 için yazılım örnekleri verilmiştir.

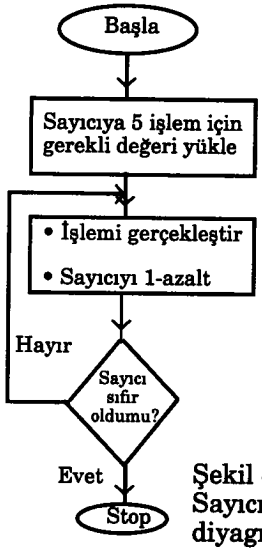
4.5 Yazılım teknikleri ve örnekleri:

Bu bölümde sayıcılar, çarpma işlemi, yığın ve kod dönüşümleri ile ilgili yazılımlar veya yazılım teknikleri gösterilmiştir.

4.5-1 Koşullu döngüler ve sayıcılar:

Sayıcılar, koşullu döngüleri açıklayan iyi bir örnektir. Örneğin; Mİ aynı işi beş defa nasıl tekrarlar? Şekil 4.7'de bu işlem için uygun bir algoritma gösterilmiştir. Bunun için, Mİ bir sayıcı kullanır ve sayım işlemi bittiği anda, bir flağın değerinin değişmesi ile işlemin bittiğini belirtir.

1) Genel amaçlı registerlerden birisine uygun bir değeri yükleyerek sayıcı oluşturulur.
2) Sayma işlemi, bu registeri 1-artırarak veya 1-azaltarak gerçekleştirilir.
3) Döngü, koşullu satır atlama komutları ile gerçekleşir.
4) Sayma sonunda bir flağın değeri değişir.



Şekil 4.7: Sayıcı için akış diyagramı

Sıfır flağını etkilemek için, aşağı doğru saymak, yukarı doğru saymaktan daha kolaydır. Bu yöntemde gösterildiği gibi, bir register çiftini bellekteki adresleri gösteren bir gösterge şeklinde kullanarak, bellek içindeki bir veri bloğu taranabilir.

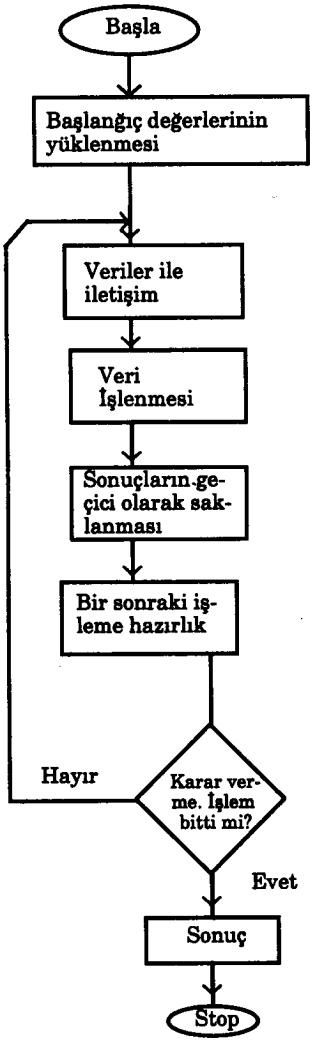
Örnek 4.1:

Bellekte, belirli bir adresten başlayarak sırayla dizilmiş on adet verinin toplamını bulup sonucu gösteren bir algoritma çiziniz.

Çözüm:

Mİ'nın şu işlemleri yapması gerekmektedir.

1) 10 verinin sayılması için, genel amaçlı registerlerden birisine 10 değerinin yüklenmesi
2) Bellekte erişilecek adresin hesaplanması ve saklanması
3) Bellekten Mİ'ye (ALU) verinin taşınması
4) Toplamanın gerçekleşmesi
5) Toplama işlemlerinin sonucunun geçici olarak saklanması
6) Sayma işleminin tamamlanıp, tamamlanmadığını flaglardan görmesi
7) Sonucun saklanması



1) Bu adımda sayıcı 8-bit registere ve belleğe erişim için kullanılacak register çiftine gerekli değerler yüklenir.

2) Sıradaki veri bellekten Mİ'ye taşınır.

3) Blokta bulunan sıradaki veri toplanır.

4) Şimdiye kadar elde edilmiş toplama sonuçları uygun bir registerde saklanır.

5) Bu noktada Mİ, dizinin bitip, bitmediğini bilemez. İşlemi yeniden tekrarlıyabilmesi için sayıcının ve bellekte erişilecek adresin değerlerinin belirlenmesi gerekir.

6) İlgili flağın aldığı değere göre karar verilir.

7) İşlem bittikten sonra sonuç, çıkış portlarından birisinde gösterilir.

Şekil 4.8: Örnek 4.1 için algoritma

4.5-2 Çarpma işlemi:

2 tabanında yapılan çarpma işlemi ile 10 tabanında yapılan çarpma işlemi tamamen aynıdır. Mİ içinde yapılan çarpma işlemini açıklarken, kolaylık sağlaması için işleme giren veriler Şekil 4-9'da gösterildiği gibi isimlendirilmişlerdir. Çarpılan veri, çarpan veri-

nin her biti ile sırayla çarpılmaktadır. Bu araçarpımlarının sonuçları, ya sıfır yada, çarpılan verinin değerini aldıklarından, 2 tabanında çarpma işlemi 10 tabanında olduğundan daha kolaydır.

```
  1011 ← Çarpılan veri
  1101 ← Çarpan veri
  ----
  1011
 0000
1011          } Araçarpım sonuçları
1011
--------
10001111  ← Sonuç
```

Şekil 4.9: 2 tabanında normal çarpma işlemi

Araçarpım sonuçlarının toplanması ile, sonuç bulunur. Kağıt kalem ile yaptığımız çarpma işleminde araçarpım sonuçlarını bulup, daha sonra bunları toplamak uygundur. Bununla birlikte, Mİ içinde yapılan çarpmada, genel amaçlı registerleri daha az kullanmak için, araçarpım sonuçları bulunduktan hemen sonra toplanırlar. Şekil 4.9'da gösterilen çarpma işlemine en sağdaki bitten başladık. Araçarpım sonucunu bulduktan sonra, bu değeri sola kaydırdık. Mİ içinde yaptığımız çarpmada bu işlemin tersini yapmak daha uygundur. Şekil 4.10'da gösterildiği gibi işleme en soldaki bitten başlanır ve araçarpım sonuçlarının toplamı kaydırılır.

```
1011        1011        1011        1011
1101        1101        1101        1101
  •           •           •           •
------      ------      ------      ------
1011        1011      100001      1000010
              1011        0000         1011
            ------      ------      -------
            100001      1000010     10001111
```

Şekil 4.10: Mİ içinde yapılan çarpma işlemi

Örnek 4.2:

Bellek adresleri verilmiş iki sayıyı çarpan bir program yazalım. Çarpılan veri 0040 H, çarpan veri 0041 H, adreslerinde yazılı olsun. Bu iki değeri çarpıp sonucu yine bellekte saklayalım. 8 bit-

lik iki sayının çarpımından elde edilen sonuç için bellekte 16 bitlik bir yer ayrılmalıdır Bu örnek programda sonuç 0042 H ve 0043 H adreslerinde saklansın.

	LXI H 40 00	HL register çiftine çarpılan verinin adresi olan 0040H değeri yüklendi.
	MOV E, M	Çarpılan veri bellekten E registerine alındı.
	MVI D 00	DE register çiftinin D bölümünün içeriği 00H yapıldı.
	INX H	Çarpan verinin adresini göstermesi için, HL register çiftinin içeriği 1-artırıldı.
	MOV A, M	Çarpan veri bellekten akümlatöre alındı.
	LXI H 00 00	HL register çiftinin içeriği 0000H yapıldı. Bu satıra kadar HL, adresleri saklamak için kullanılmıştı. Bundan sonra, araçarpım sonuçlarını saklamak için kullanılacak.
	MVI B 08	B registerine 8 değeri yüklendi. Bu register, çarpan verinin bütün bitlerini işleme sokmak için, sayıcı olarak kullanılacak.
ÇARP	DAD H	HL register çiftinin içeriği kendisi ile toplanır. Bu işlem, içeriği 2 ile çarpma işlemine denktir. Böylece HL register çiftinin içeriği 1-bit sola kaymış olur. Başlangıçta bu içerik 0000H olduğundan, komutun ilk işlenmesinde hiçbir etkisi yoktur.
	RAL	Elde flağı üzerinden akümlatör sola kaydırılır. Böylece çarpan verinin MSB'i elde flağına alınmış olur.
	JNC SAYDE	Eğer elde flağının değeri 0 ise, 3-baytlık bu komut ile SAYDE adresine atlanır. Elde flağının değeri 1 ise, sıradaki komut işlenir.

```
        DAD D       DE register çiftinin içeriği ile HL register çiftinin içeriği toplanır. D registerinin içeriği 00H'dir ve E registerinde çarpılan veri bulunmaktadır.

SAYDE   DCR B       B registerinin içeriği 1- azaltıldı.

        JNZ         Eğer B registerinin içeriği 00H değil ise, bu
        ÇARP        3 baytlık komut ile, ÇARP adresinde yazılı komuta atlanır. B registerinin içeriği 00H değerine inmiş ise, sıradaki komut işlenir.

        SHLD        Sonuç 0042H ve 0043H adreslerine yazılır.
        42
        00

        HLT         Mİ bu satırda durur.
```

Mİ içinde bölme işlemini gerçekleştirmek için bölen sayı, bölünen sayıdan tekrarlı olarak çıkarılır. Bu çıkarma işlemi artan sayı 0 veya negatif olana kadar sürdürülür. Genel amaçlı registerlerden bir tanesi, her çıkarma işleminde 1-artırılır. Çıkarma işlemine son verildiği anda, bu genel amaçlı registerde, bölme işleminin sonucu bulunur.

Örnek 4.3:

Kritik bir notaya, uyarı amacı ile iki adet ışık yerleştirilecektir. Bu ışıklar 1sn'lik aralıkla yanıp söneceklerdir. Işıklardan birisi yanık durumdayken, diğeri sönük durumda olacaktır. Bu sistem, saat sinyalinin periyodu 0.5μs olan bir Mİ ile kontrol edilecektir.

Program:

```
                MVI D, AAH      D registerine 10101010 değeri yüklendi.
DÖN             MOV A, D        Bu bit dizisi akümlatöre aktarıldı.
                RLC             Akümlatörün içeriği AAH veya 55H olarak değiştirilir.
                MOV D, A        Akümlatörün içeriği saklandı.
```

	ANI 03 H OUT PORT 1	D7-D2 bitleri maskelendi. Işıkları açıp, kapayan sinyaller PORT1'e yazıldı.
	LXI B, SAYICI	BC register çiftine SAYICI değeri yazıldı. Bu değer yardımı ile, bu Mİ için 1sn'lik bir bekleme süreci oluşturulacaktır.
BEKLE	DCX B	SAYICI değeri 1-azaltıldı.
	NOP NOP	Bekleme döngüsünü uzatmak için, işlemi değiştirmeyen saat sinyali periyodları eklendi
	MOV A, C ORA B	Sıfır flağını etkilemek için B registeri VEYA işlemine sokuldu.
	JNZ BEKLE	Eğer BC register çiftinin içeriği sıfır değil ise, bekleme döngüsüne devam edilir.
	JMP DÖN	Bu satıra gelince BC register çiftinin içeriği sıfır olmuş ve 1sn'lik süre geçmiş demektir. Yeni dizi için DÖN satırına atlanır.

4.5-3 Yığın:

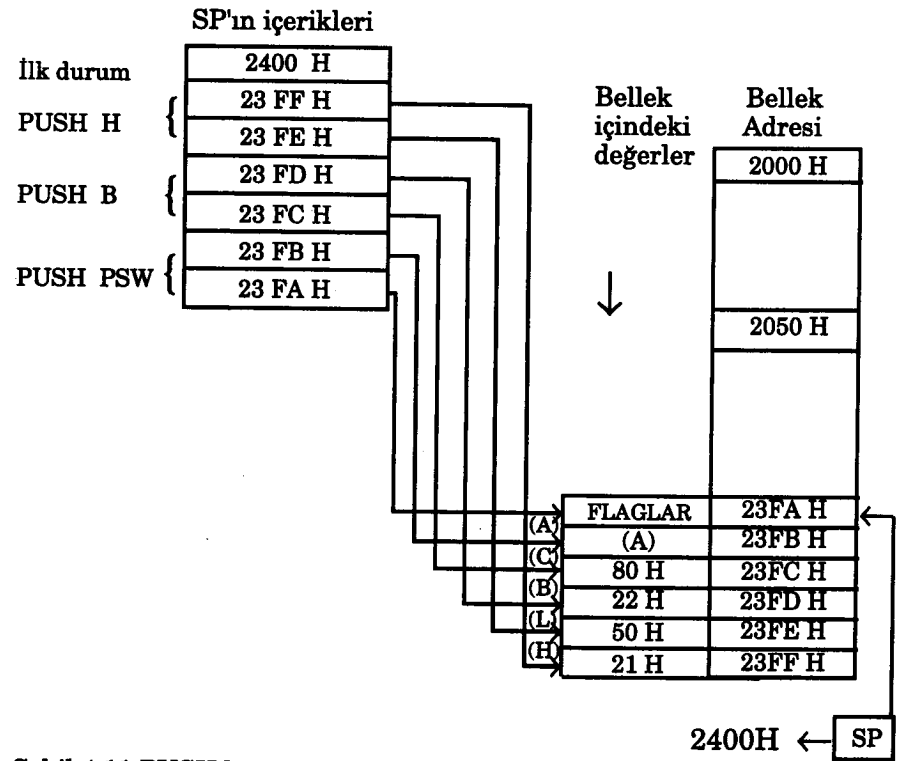
Örnek 4.4: Bellek içinde, kullanabileceğimiz bölge 2000H ve 23FF H adresleri arasındadır. 2000H ve 2050 H adresleri arası, veri transferi ve aritmetik işlemler için kullanılmaktadır. SP'a 2400 H adresi yüklenmiştir. 2150 H ve 2280 H adreslerinden başlayan iki adet veri dizisi bulunmaktadır. HL ve BC register çiftleri bellekte erişilecek adresleri saklamak için kullanılacaklardır. Aşağıdaki programı inceleyiniz.

Çözüm:

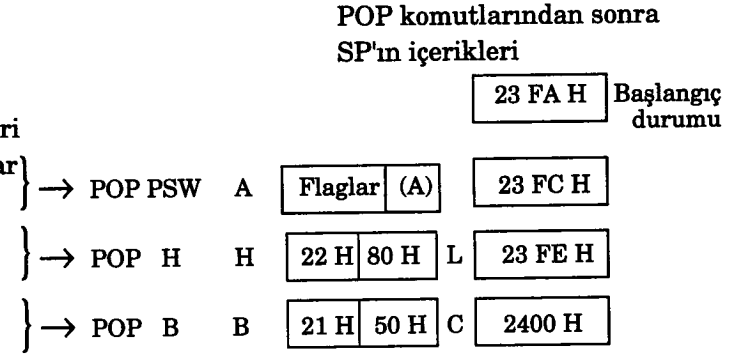
```
2000 H   LXI SP, 2400 H
2003 H   LXI  H, 2150  H
2006 H   LXI  B, 2280  H
2009 H   MOV  A, M
200A H   PUSH H
200B H   PUSH B
200C H   PUSH  PSW
200D H
            ↓
2020 H   POP  PSW
2021 H   POP  H
2022 H   POP  B
```

Registerlerin İçerikleri

	Flaglar	Veri
A		
B	22 H	80 H
D	xx	xx
H	21 H	50 H
SP	2400 H	



Şekil 4.11 PUSH komutunun işlenmesinden sonra yığının içerikleri



Şekil 4.12: POP komutunun işlenmesinden sonra registerlerinin içerikleri

Programın başlangıcında SP'a 2400 H adresi yazılmıştır. Yığının gerçekte başlangıç adresi, bu adresin 1-üstü olduğundan bu uygulama doğrudur. PUSH komutu işlenirken ilk önce SP'ın içeriği 1-azaltılır, daha sonra veri yığına aktarılır. Şekil 4.11'de üç adet PUSH komutu işlendikten sonra, yığın ve SP'ın içerikleri gösterilmiştir. PUSH (H, B ve PSW) komutlarının işlenmesinden sonra yığın yukarı doğru büyümüştür. PUSH komutları kullanılırken yığının programla karışacak kadar büyümemesine dikkat edilmelidir. Şekil 4.12'de POP komutlarından sonra registerlerin içerikleri gösterilmiştir. Yığından veri alınırken, ilk alınan veri, yığına son yazılan veridir. Eğer registerlerin içerikleri yığına aktırıldıktan sonra, yeniden yığından aynı registerlere yazılacak ise, bu kural gözönünde alınmalıdır. Yukarıdaki programda registerlerin yığına aktarılmasında HL, BC ve PSW sırası izlenmiştir. Eğer registerlere aynı değerler yeniden yazılacak ise, PSW, BC ve HL dizisi izlenmelidir. Bu örnekte, dizi değiştiğinde oluşacak farkı göstermek için PSW, HL ve BC dizisi izlenmiştir.

4.5-4 BCD sayıların 2 tabanında yazılması:

Mİ kontrollü birçok sistemde, sisteme girilen veriler, bir tuştakımı yardımı ile BCD formda girilir. Aynı şekilde sistem içinde bulunan sonuçlar göstergeye yazılmadan önce BCD forma sokulurlar. Mikrobilgisayar sisteminin sistem programı, her tuşun atandığı değeri, 4-bit içinde, ikilik tabanda belirler ve iki adet BCD değeri, 8-bitlik bir registerde veya bellekte saklar. Bu gösterime paketli BCD form denir. Sisteme veriler BCD formda girilse bile, bu veriler işlenirken, 2-tabanında gösterilmelidir.

2 basamaklı bir BCD sayının, 2-tabanında yazılması için şu işlemler yapılır.

1) 8-bitlik paketli formdaki BCD sayı, BCD_1 ve BCD_2 olmak üzere iki adet paketsiz BCD sayı şeklinde yazılır.

2) Her iki BCD sayı da, bulunduğu basamağa göre 2-tabanında yazılır.

3) Başta verilen BCD sayının, 2-tabanındaki değerini bulmak için, 2. adımda bulunan sayılar toplanır.

Örnek 4.5:

$(72)_{BCD}$ olarak verilen sayının, iki tabanındaki değerini hesaplayınız.

Çözüm:

72=70x10+2x1

72_{10}= 0111 0010_{BCD}

1. adım: 0111 0010→ 0000 0010 paketsiz BCD_1

→ 0000 0111 paketsiz BCD_2

2. adım: BCD_2 10 ile çarpılmalı: (7x10)

3. adım: BCD_1 ile 2. adımın sonunda bulunan değer toplanmalı.

Örnek 4.6:

0 ile 99 arasında bir sayı, BCD formda, giriş portunda yazılıdır. Bu sayıyı 2-tabanında yazıp, sonucu çıkış portuna gönderen bir program yazınız.

Çözüm:

BAŞLA	LXI SP,	YIĞIN	SP'a başlangıç değeri yazıldı.
	IN	Gİ-PORT	BCD sayı Mİ'ye alındı.
	CALL	BCD-İKİ TAB	BCD sayıyının iki tabanındaki değerini bulan altprograma atlandı.
	OUT	ÇI-PORT	2 tabanındaki değer çıkış portuna yazıldı.
	HLT		Programın sonu.

BCD-İKİ TAB: Bu altprogram paketli BCD sayının 2-tabanındaki değerini bulur.

	PUSH B	BC register çifti saklanır.
	PUSH D	DE register çifti saklanır.
	MOV B, A	BCD sayı saklanır.
	ANI 0F H	Yüksek mertebeli 4-bit maskelenir.
	MOV C, A	Paketsiz BCD_1, C registerinde saklanır.
	MOV A, B	BCD sayı yeniden getirilir.
	ANI F0H	Düşük mertebeli 4-bit maskelenir.
	RRC	Yüksek mertebeli 4-bit, paketsiz BCD_2 şekline sokulur.
	RRC	
	RRC	
	RRC	
	MOV D, A	BCD_2 D registerinde saklanır.
	XRA A	Akümlatörün içeriği 00H yapılır.
	MVI E, 0AH	E registerinde 10 çarpanı saklanır.
TOP	ADD E	D=0 olana kadar, 10 değeri toplanır.
	DCR D	BCD_2 1-azaltılır.
	JNZ TOP	Çarpma işlemi tamamlanmamış ise, toplama işlemine devam edilir.
	ADD C	BCD_1 toplanır.
	POP D	Registerlere eski değerleri aktarılır.
	POP B	
	RET	

4.5-5 2 tabanındaki sayıların BCD formda yazılması:

Mİ içinde veriler işlenirken, sayılar 2-tabanında yazılmışlardır. Bulunan sonuçlar ise, gösterge birimine gönderilmeden önce BCD forma alınmalıdır. Bu dönüşümde, 2-tabanındaki sayı 10'nun kuvvetlerine bölünür. Bu bölme işleminde ise çıkarma yöntemi kullanılır.

Örneğin ilgilendiğimiz sayı

$1111\ 1111_2 = FFH = 255_{10}$

olsun.

Bu sayının BCD formda yazılabilmesi için BCD_3 (MSB), BCD_2 ve BCD_1(LSB) olmak üzere 12 bite gereksinim vardır.

= 0010 0101 0101
BCD_3 BCD_2 BCD_1

Dönüşüm işlemi için şu adımlar gereklidir.

1. Adım

Eğer sayı 100'den küçük ise 2. adıma geçin. Diğer durumda, artan 100'den küçük olana kadar 100 eksiltin. Çıkarma işlemi sayısı BCD_3 olacaktır.

Örnek

255
-100 = 155 1
-100 = 55 1
BCD_3 = 2

2. Adım

Eğer sayı 10'dan küçük ise 3. adıma geçin. Diğer durumda, artan 10'dan küçük olana kadar çıkarma işlemine devam edin

55
-10 = 45 1
-10 = 35 1
-10 = 25 1
-10 =15 1
-10 = 5 1
BCD_2 = 5

3. Adım

2. adımdan elde edilen artan BCD_1 olur.

BCD_1 = 5

Bu işlemi gerçekleştiren program aşağıda verilmiştir. Bu program 2-tabanındaki sekiz bitlik sayıyı BCD forma çevirdiğinden, üç adet BCD basamak için, 12 bite gereksinim vardır. Sonuç bellekte paketsiz BCD formda saklanır.

BAŞLA	LXI SP, YIĞIN	SP'a başlangıç adresi yüklendi.
	LXI H, 2-BAYT	2-tabanındaki sayıyı göstermesi için, HL register çiftine gerekli adres yüklendi.
	MOV A, M	2 tabanındaki sayı akümlatöre alındı.
	CALL ON	10'un kuvvetlerini yüklemek için altprograma geçildi.
	HLT	
ON	LXI H, ADRS	HL register çiftinin bellekte ilk BCD basamağın yazılacağı adresi göstermesi sağlandı.
	MVI B, 64 H	B registerine 100 sayısı yüklendi.
	CALL DÖNÜŞ	Dönüşüm altprogramına geçildi.
	MVI B, 0AH	B registerine 10 sayısı yüklendi.
	CALL DÖNÜŞ	Dönüşüm altprogramına geçildi.
	MOV M, A	BCD_1 yazıldı.
	RET	
DÖNÜŞ	MVI M, FF	(0-1) değeri belleğe yazıldı.
DÖNGÜ	INR M	Bu adresin içeriğinin 00H olması ve bölme işlemindeki çıkarma sayısının aynı adrese yazılması sağlandı.
	SUB B	2-tabanındaki sayıdan 10'nun kuvveti çıkarıldı.
	JNC DÖNGÜ	Eğer kalan sayı 10'nun kuvvetinden büyük ise belleğin içeriği 1-artırılır.

ADD B	Eğer kalan sayı 10'nun kuvvetinden küçük ise, artanı bulmak için 10 nun kuvveti eklenir.
INX H	HL register çiftinin bellekte bir sonraki adresi göstermesi sağladı.
RET	

Anaprogram, BCD forma girecek veriyi akümlatöre alır ve ON altprogramını çağırır. ON altprogramı sonucun saklanacağı bellek adresini belirler ve B registerine, 10'un kuvvetlerini yükler. Bundan sonra DÖNÜŞ altprogramını çağırır. DÖNÜŞ altprogramında, bellekte gerekli adresin içeriği, çıkartma sayısı kadar 1-artırılır. Bu altprogram, birkez B registerine 64 H ve birkez de 0A H yüklendikten sonra, toplam iki kez çağrılır. Her iki çağrımda da ilk önce ilgili adresin içeriği, 00H olur ve daha sonra hesaplanan değer bu adreste saklanır. DÖNÜŞ altprogramı aynı zamanda HL register çiftinin, hesaplanan BCD formdaki değerin, bellekte yazılacağı adresi göstermesini sağlar.

YIGIN = 4000
2-BYTE = 2000
ADRS = 2100

2000 Adresine yazılı (00-FF) 2 tabanlı sayı
2100, 2101, 2102 Adresine BCD ye çevrilecek yazılır
örnek 2000 Adr FF → 2100 2101 2102
02 05 05 255
0A → 00 01 00 10

2000 Adr
$55_H = 0101\ 0101 = 1+4+16+64 = 85_{10}$ → 85_{BCD} 2100 → 0 1
$66_H = 0110\ 0110 = 64+32+4+2 = 102_{10}$ → 2101 → 8 0
2102 → 5 2

$55_H = 5 \times 16 + 5 = 85_{10}$
$66_H = 6 \cdot 16 + 6 = 102_{10}$

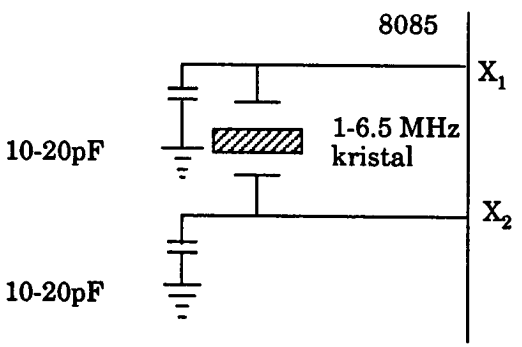
BÖLÜM V
Intel 8085'in Donanım Yapısı

Intel 8085, kullanımı kolay ve genel amaçlı 8-bitlik bir Mİ'dir. Yazılım açısından 8080 ile uyumludur. Bundan başka 8085 komut listesinde RIM ve SIM komutları da bulunur.

5.1 Genel mimari:

5.1-1 Osilatör devresi:

8085 içinde ek olarak yanlız bir kristalin bağlanmasını sağlıyan bir osilatör devresi vardır. Bu kristalin bağlantısı Şekil 5.1'de gösterilmiştir. Mİ kendi saat sinyalini üretmek için, bu kristalin frekansını ikiye böler. Örneğin 8085'in kusursuz çalışması için saat sinyalinin periyodunun en az 320 nsn olması gerekir. Bu periyod ise 3.125 MHz'e karşılık gelmektedir. Eğer Mİ'nin bu hızda çalışması isteniyorsa, 6.25 MHz'lik bir kristal bağlanması gerekir. Bundan başka saat sinyalinin periyodu en çok 2μsn olabilir ve buda 500kHz'e karşılık gelir. Dolayısı ile 8085 bağlanacak kristalin frekansı en az 1MHz olmalıdır.



Şekil 5-1: 8085'e kristalin bağlanması

Intel 8085 ile çalışırken kristal devresinden kaynaklanabilen bir soruna karşı hazırlıklı olmamız gerekir. Mİ'nin içinde bulunan saat sinyali osilatörü, kullanılan kristalin üçüncü harmoniğinde çalışmaya başlıyabilir. Bu durum, genellikle kristal frekansı 5

MHz'in üzerinde seçilmiş ise görülür. Bu sorunun çözülmesi için en iyi yöntem kristale kapasitör bağlamaktır. Bu güne kadar elde edilen deneyimler sonucu, 10 pF ile 20pF değerleri arasındaki iki kapasitörün, Şekil 5-1'de gösterildiği gibi toprak ile kristalin uçları arasına bağlanması en iyi çözümdür.

5.1-2 Veri yolunun paylaşımı:

Intel 8085'in tümdevresi üzerindeki bacak sayısını minumum sayıda tutmak için, bazı bacaklara birden fazla görev verilmiştir.

Bu Mİ'de, adres yolunun düşük mertebeli 8 biti ve veri bitleri aynı hatlar üzerinden aktarılırlar. Bu hatlara ilk önce bellek adresi yazıldıktan sonra, bir sonraki çevrimde veri yazılır. Bu hatlar veriyi çift yönlü olarak taşıyabilirler. Intel 8085'in ALE (Address Latch Enable) sinyali, bu hatlar adresi gösterirken lojik 1 seviyeye çıkar.

8085'in diğer kontrol sinyalleri ise aşağıda, açıklanmıştır.

$\overline{RD}$: Mİ, bellekten veya bir çevre elemanından veri okurken, bu hat lojik 0 seviyeye iner.

$\overline{WR}$: Belleğe veya bir çevre elemanına veri yazılırken, bu çıkış lojik 0 seviyeye iner.

IO/$\overline{M}$: Bu hat iletişim kurulan birimin, bellek (IO/$\overline{M}$=0) veya çevre elemanı (IO/$\overline{M}$=1) olduğunu belirtir.

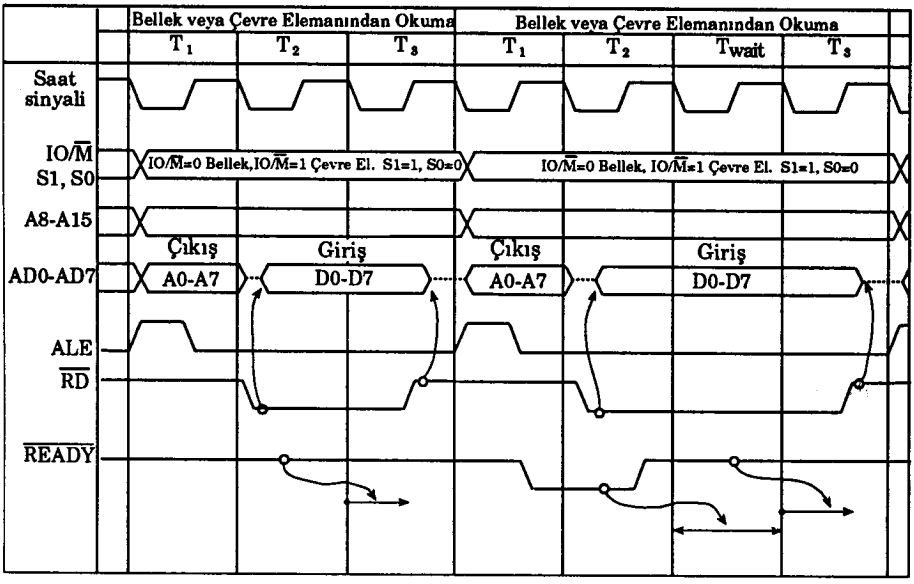
S_1,S_0: Bu hatların çıkışındaki kod çözülerek bellek işleminin türü anlaşılır. (I/0 okuma, belleğe yazma, opcode getirilmesi, vb.) Genellikle 8085 ile tasarlanan devrelerde bu bilgilere gereksinim duyulmaz.

$\overline{READY}$: Bu hat Mİ için bir giriş hattıdır. Her makina çevriminin 2. periyodunda bu girişin lojik seviyesi Mİ tarafından belirlenir. Eğer $\overline{READY}$ giriş lojik 0 seviyede ise, Mİ bu giriş lojik 1 seviyeye çıkana kadar işlem yapmaz. Bu hat düşük hızlı bellek devreleri ile Intel 8085 arasında uyum sağlamak için kullanılır.

Kullanılan bellek tümdevresinin hızı 8085'e göre çok düşük ise, belleğe bir sözcük yazılana kadar veya bellekten bir sözcük okunana kadar bu giriş lojik 0 seviyede tutulur.

8085'in kontrol yolunun çalışması Şekil 5.2'de gösterilmiştir. Gösterilen iki okuma çevriminden birincisinde BEKLEME duru-

mu ($\overline{\text{READY}}$, lojik 1) yoktur. İkinci çevrimde ise bir BEKLEME durumu ($\overline{\text{READY}}$ lojik 0) vardır. AD0-AD7 (adres ve veri için ortak kullanılan hatlar) hatlarında, ilk önce T1 sırasında adres görülüyor ve daha sonra T2 sırasında veri hazır oluyor.



Şekil 5.2: Bellekten veya bir çevre elemanından OKU işlemi sırasındaki makina çevrimleri.

Örnek 5.1:

Eğer 8085 ile kullanılacak bellek tümdevresinin hızı yeteri kadar yüksek değil ise, her bellek çevriminde $\overline{\text{READY}}$ girişinin lojik 0 seviyeye düşmesi nasıl sağlanır?

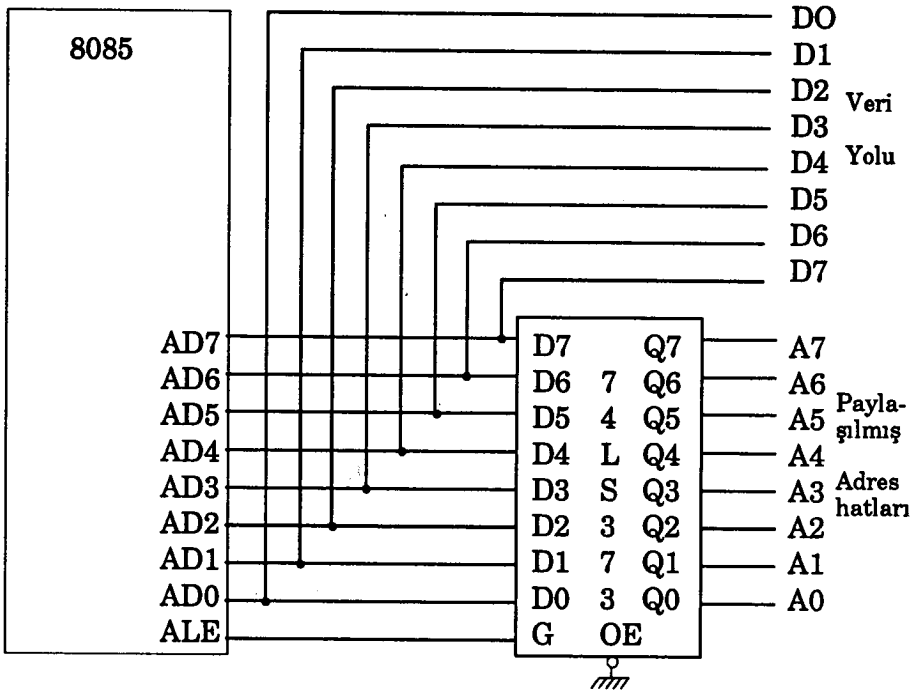
Çözüm:

Her bellek çevrimi başlangıcında görülen ALE sinyali, bir one-shot devresini tetiklemek için kullanılır. One-shot devresindeki palsın uzunluğu Mİ ile bellek tümdevresi arasında uyumu sağlıyacak şekilde ayarlandıktan sonra, bu devrenin çıkışı Mİ'nin $\overline{\text{READY}}$ girişine bağlanır.

Örnek 5.2:

Intel 8085 ile tasarım yaparken veri ve adres hatlarının paylaşımını gerçekleştiren bir devre öneriniz.

Çözüm:



Mİ'ye bir bellek tümdevresi bağlanmadığı veya bellek tümdevresinin bozulmuş olduğu durumlarda, bir OKU çevrimi sırasında, Mİ adres yolunun düşük mertebeli 8 bitini veri olarak algılar. Böyle bir durumda, Mİ'ye besleme voltajı ve RESET sinyali uygulandığında karşılaşılan sonuç aşağıda verilmiştir.

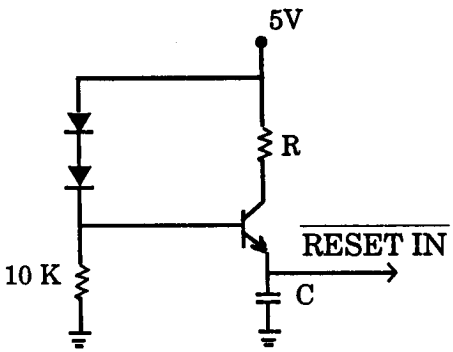
Adres	Veri	Komut
0000	00	NOP
0001	01	LXI B
0002	02	STAX B
0003	03	INX B
0004	04	INR B
0005	05	DCR B

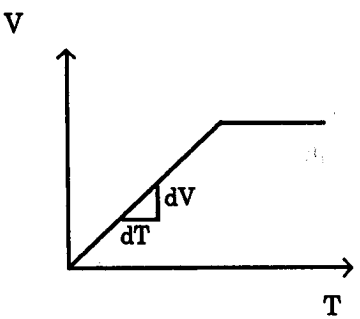
Mİ bu konumda çalışmayı sürdürürken 76 H adresinde HALT komutunu okur ve bu adreste çalışmasını durdurur.

5.1-3 8085'te bulunan diğer hatlar:

Bu bölümde RESET IN, RESET OUT ve CLOCK OUT sinyalleri açıklanacaktır.

RESET IN: Bu girişe lojik 0 seviyede bir sinyal uygulandığı durumda Mİ reset duruma geçer. Mİ'ye uygulanan besleme voltajı 5V'ta ulaştıktan sonra RESET IN sinyali en az 10 ms süresince lojik 0 seviyede kalmalıdır. Bu sinyal uygulandıktan sonra PC'ın ve diğer registerlerin içerikleri 00H olur. Bu sinyali üretebilen devre Şekil 5.3 te verilmiştir.





$$\frac{dV}{dT} = \frac{0.6}{CR}$$

$dT = 112\ \mu s$

$dV = 1.55\ V$

$$13900 = \frac{0.6}{CR}$$

Şekil 5.3: Besleme voltajı verildiğinde, RESET sinyalini üreten devre.

Tablo 5.1: $\overline{\text{RESET IN}}$ sinyalinin 8085'e etkisi

Devre	Reset/Set
Program Sayıcısı	Reset
Komut registeri	Reset
INTE flip-flop	Reset
RST 7.5 flip-flop	Reset
TRAP flip-flop	Reset
SOD flip-flop	Reset
RST 5.5 mask	Set
RST 6.5 mask	Set
RST 7.5 mask	Set
HOLD, INTR ve READY için içteki flip-floplar	Reset

$\overline{\text{RESET}}$ OUT: Aktif durumu lojik 1 olan bu çıkış, 8085'e gelen $\overline{\text{RESET IN}}$ sinyalini sistemde bulunan diğer tümdevrelere iletmek için kullanılır. Bu çıkış yardımı ile, sistemdeki diğer elemanların Mİ ile aynı anda reset edilmesi sağlanır. 8085'i $\overline{\text{RESET IN}}$ girişine lojik 0 seviye uygulanması ile RESET OUT çıkışı lojik 1 seviyeye gelir.

CLOCK OUT: 8085'in bu çıkışında, kristal frenkansının yarısı değerde bir kare dalga bulunur. Bu kare dalga, sistemde, Mİ ile uyum içinde çalışması gereken diğer devrelerin saat sinyali olarak kullanılır.

5.2 Giriş ve çıkış portları:

Intel 8085 Giriş/Çıkış komutları, giriş ve çıkış portları üzerinden gerçekleşir. Bir port IN veya OUT komutlarında verinin kaynağını belirler. Örneğin Mİ'ye veri gönderen birimler giriş portları-

na ve Mİ'den veri alan birimler çıkış portlarına bağlanmalıdır. 8085'in donanım yapısı ile bu Mİ'ye 256 giriş ve 256 çıkış portu bağlanması mümkündür.

IN ve OUT komutları giriş ve çıkış işlemlerini gerçekleştirir. IN komutu ile bir çevre elemanından akümlatöre veri alınırken, OUT komutu ile akümlatörün içeriği bir çevre elemanına yazılır. Her iki komut için de bellekte iki adet register kullanılır. Komutun birinci baytı opcode ve ikinci baytı port adresidir. (256 porttan herhangi birisi seçilebilir.) IN komutunun opcode'u DBH ve OUT komutunun opcode'u D3H'tir. Mİ opcode'u okuduktan sonra ilgili port adresini okur.

Bu iki çevrimin arkasından veri çıkışı veya girişi gerçekleşir. Komutun ikinci baytında verilen 8-bitlik port adresi 16-bitlik adres yolunun iki bölümüne de yazılır. Aynı adres hem 0-7 numaralı, hemde 8-15 numaralı adres hatlarında görülür.

Bir IN komutunun işlenme safhasında $\overline{RD}$ çıkışı lojik 0 seviyeye iner ve ilgili çevre elemanından alınan veri akümlatöre yazılır. Bu nedenle IN komutu bazı durumlarda **IOR (I/0 READ)** olarak adlandırılır. Bir OUT komutunda ise $\overline{WR}$ çıkışı lojik 0 seviyeye düşer ve akümlatörün içeriği porta yazılır. Bu işleme de kısaca **IOW (I/O WRITE)** denir.

IN ve OUT komutlarında, sistemin içinde bulunan diğer birimlere, yapılan işlemin bellek çevrimi yerine Giriş/Çıkış çevrimi olduğunun bildirilmesi gerekir. 8085'te bu işlem IO/M çıkışı ile bildirilir. 8085 komut listesi içindeki komutlardan yanlızca IN ve OUT komutları $\overline{IO/M}$ çıkışını etkiler.

5.2-1 IN ve OUT zamanlaması:

8085 Mİ'sinin çalışması sırasında, bir OUT komutu işlenirken ilgili çıkışların değişimi Şekil 5.4'te gösterilmiştir. IN ve OUT komutları, 10 saat periyodunun içinde gerçekleşen üç adet çevrim içerirler. Şekil 5.4'te saat sinyali, ALE, $\overline{RD}$, $\overline{WR}$, A8 ve paylaşılmış adres/veri yolunun düşük mertebeli 8 biti, OUT 15 komutu için gösterilmiştir. Komut ile ilgili adımlar aşağıda açıklanmıştır.