BÖLÜM 5. KOMUT SETİ MİMARİSİ

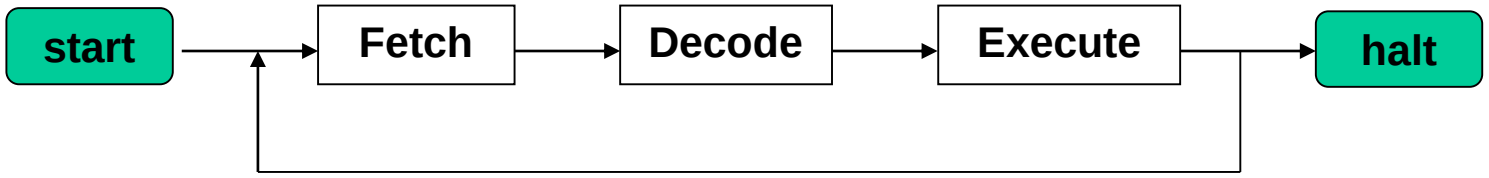
Komut Saykılı

Zamanlama Düzeneği
Komut Saykılının Fetch ve Decode Safhaları
Komut Saykılının Execute Safhası

ADD Komutunun İvedi, Direkt ve Dolaylı Mod için Mikroişlem Adımları

Komut Saykılı

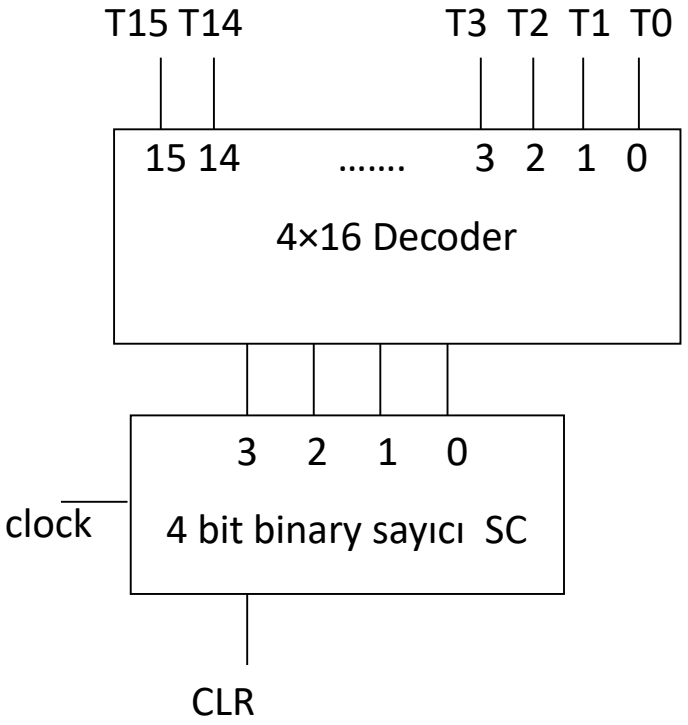
Bilgisayarın bellekte bulunan komutları sırayla işlemesi beklenir. Bir komutun yerine getirmesi gereken işlemlerin bütününe komut saykılı adı verilir. Komut saykılı 3 aşamadan oluşur bunlar; fetch (al-getir), decode (çöz) ve execute (icra et) safhalarıdır. Komutun opcode kısmının bellekten alınıp komut kaydedicisine (IR) koyulması birlikte, adresleme modu ile ilgili kısmı adresleme modu kod çözücüsüne, işlem tipi ile ilgili kısmı da komut kod çözücüsüne giriş olarak uygulanır. Daha sonra bu kod çözücülerden alınan sinyaller, ilgili komut için gerekli alt işlemlerin (mikroişlem) başlatılmasını sağlayan düzeneği aktif hale getirir.



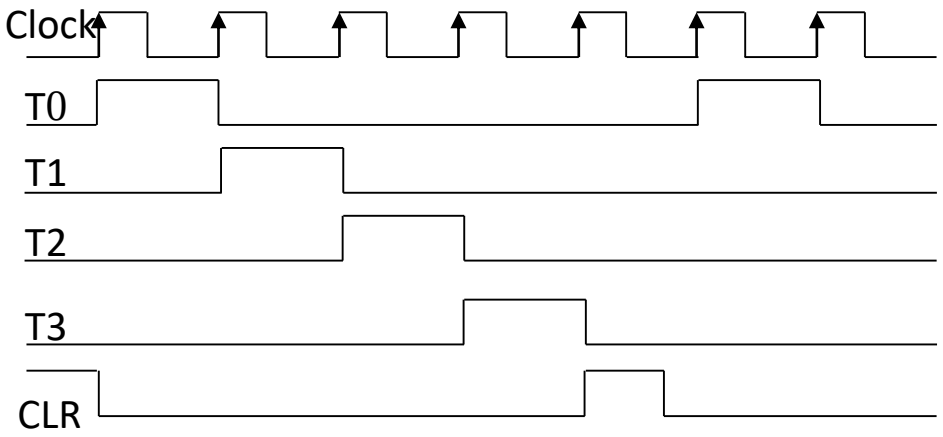
- Komut saykılı -

Zamanlama Düzeneği

Bir bilgisayar sisteminden, bellekteki komutları ve her bir komutun da sahip olduğu mikroişlemleri sıra ile işletmesi beklendiğinden dolayı donanımsal bir düzeneğe ihtiyaç vardır.



- Zamanlama düzeneği -



- Zamanlama Sinyalleri -

Komut Saykılının Fetch ve Decode Safhaları

Komuta ait alt işlemler, komutun tipine bağlı olarak farklı sayıda adımlardan oluşur. Ama tüm komutların fetch ve decode safhaları aynı yapıda olup 3 adımdan oluşmaktadır. Bu adımların zamanlaması bir sayıcı vasıtasıyla yerine getirilir. Fetch safhası iki adımda yerine getirilir (T0 ve T1). Decode safhası tek adım olup, T2 zaman diliminde yerine getirilir. Execute safhası ise T3 zaman diliminden başlayıp, komutun tipine bağlı olarak çeşitli sayıda adımdan oluşur. Ancak, doğal adresleme moduna sahip komutlar, 1 byte'lık komutlar olduğundan T2 zaman dilimindeki işlemler yapılmayacaktır.

T0 : AR ← PC

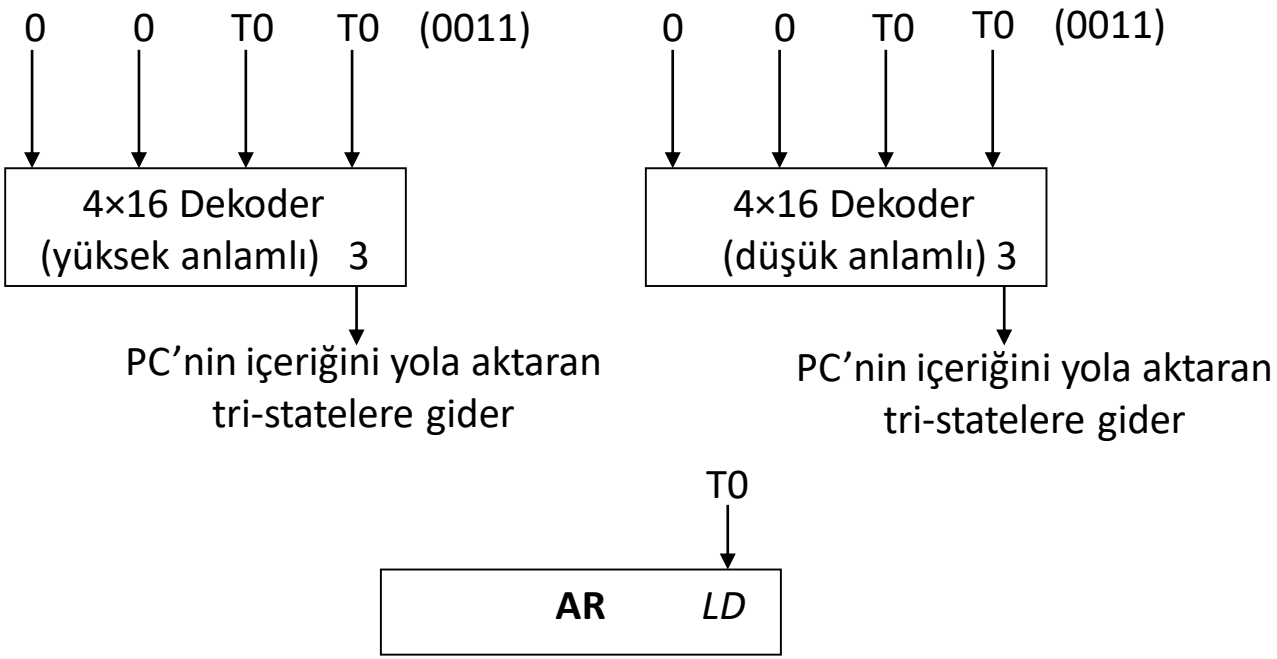
T1 : IR ← M[AR], PC ← PC+1

T2. $\overline{ADRMD0}$: AR←PC, PC ← PC+1, kod çözme aşaması

T0 Zaman Dilimi

T0 zaman diliminde PC aracılığıyla bir sonraki komutun adresi, adres kaydedicisine yüklenir. Bu işlem için, adres kaydedicisinin *LD* girişi aktif edilir, PC'nin içeriği ise ortak yola aktarılır.

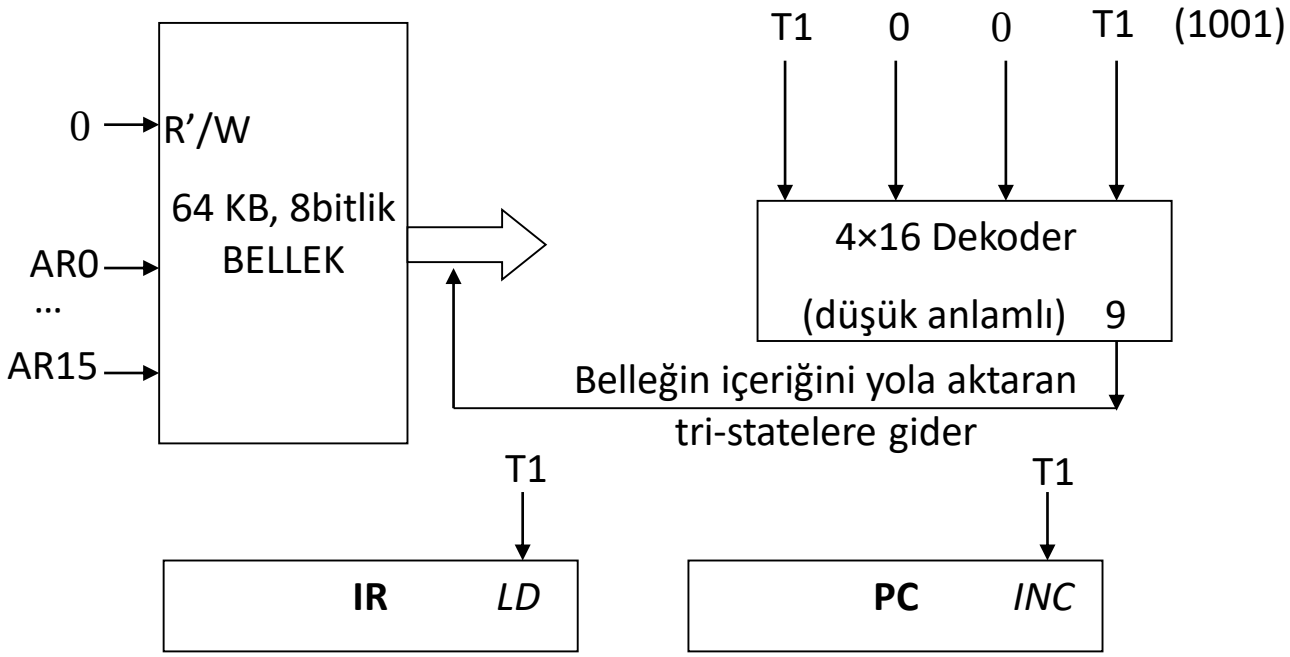
T0: AR ← PC



T1 Zaman Dilimi

T1 zaman diliminde, adres kaydedicisi ile belirtilen bellek bölgesindeki komutun opcode'u ortak yola aktarılır. Komut kaydedicisinin *LD* girişi aktif edilerek opcode, komut kaydedicisine alınır. Program sayacının içeriği 1 arttırılır (Doğal adresleme modu kullanılıyorsa bir sonraki komutu veya diğer adresleme modlarında operandın adres bilgisini göstermek için).

T1: IR ← M[AR], PC ← PC+1

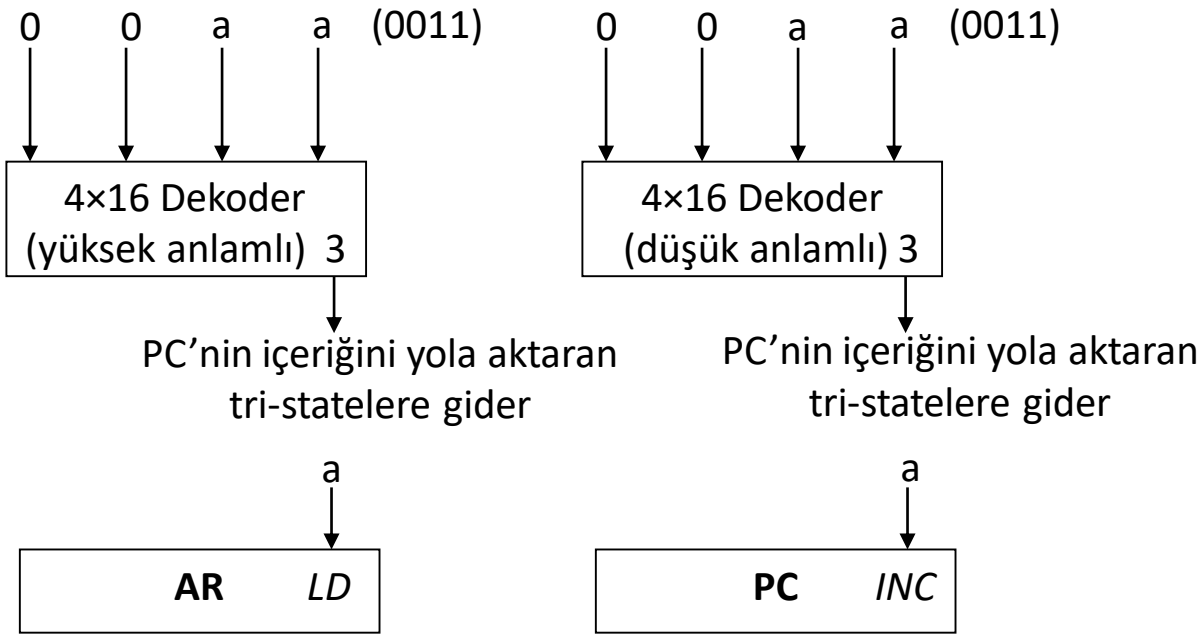


T2 Zaman Dilimi

T2 zaman diliminde, PC'nin içeriği AR'ye aktarılır. PC ise yine 1 arttırılır; şayet 2 byte lık bir komut ise bir sonraki komutu göstermek için, ya da 3 byte lık bir komutsa, komutun adres kısmındaki düşük anlamlı adres bilgisini göstermek için.

T2. $\overline{ADRMD0}$**: AR←PC, PC ← PC+1, kod çözme aşaması**

a

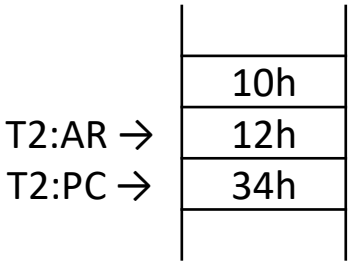


Komut Saykılının Execute Safhası

Daha önce de bahsedildiği gibi, execute safhasının kaç adım süreceği komutun tipine ve adresleme moduna bağlı olarak değişiklik göstermektedir.

Örnek: İvedi adresleme moduna sahip ADD komutunun mikroişlem adımları.

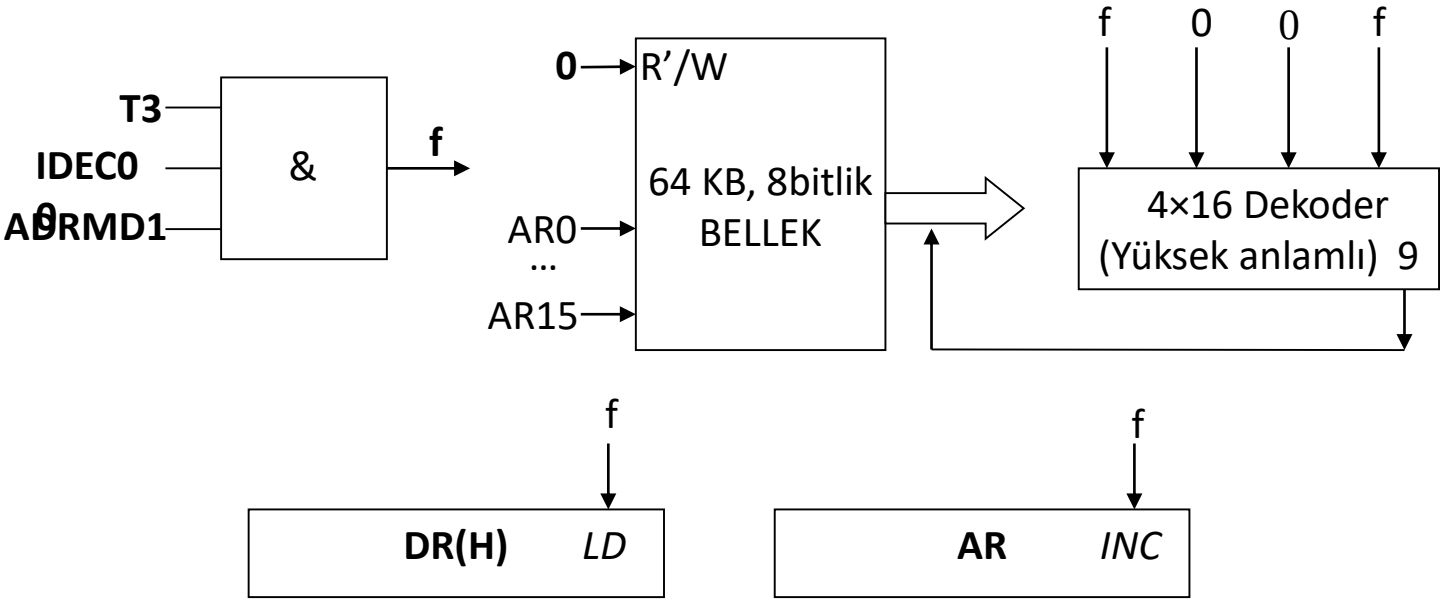
ADD #1234h komutu gibi. Burada # işareti ivedi modu göstermektedir. 3 byte'lık bir komuttur. ADD komutunun opcode'u $0\ 001\ 0000_2 = 10h$

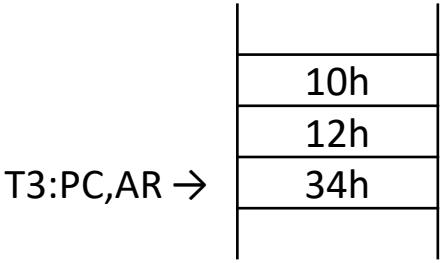


İvedi adresleme modunda akü, sabit bir sayı (1234h) ile işleme giriyordu. Akü ile bellekteki sayı toplanacak ve sonuç aküye aktarılacaktır. Şu anda AR kaydedicisi bellekteki sayımızın yüksek anlamlı kısmını göstermektedir. Bu değeri, DR kaydedicimizin yüksek anlamlı kısmına koymamız gerekecektir. Daha sonra da sayımızın düşük anlamlı kısmına erişebilmemiz için AR kaydedicisini 1 arttırmamız gerekecektir.

İvedi Mod için T3 Adımı

T3. IDEC00.ADRMD1: $DR_H \leftarrow M[AR], AR \leftarrow AR+1$

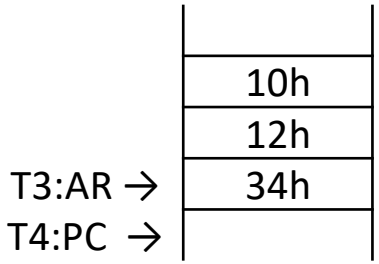




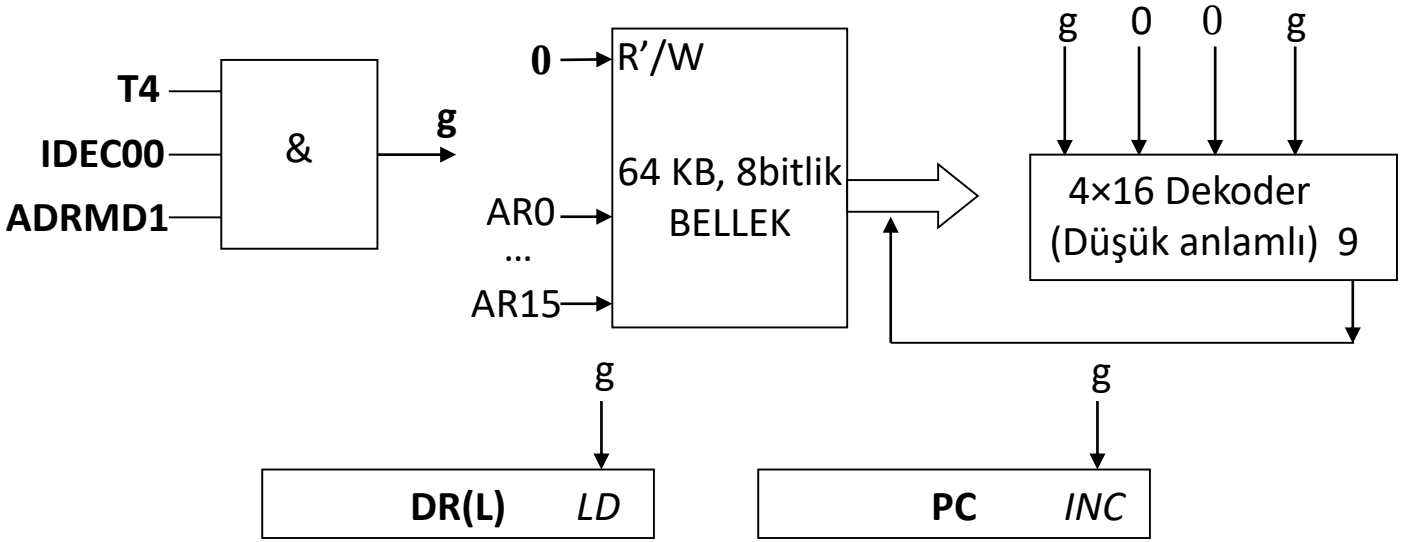
T3 adımında AR ← AR+1 işlemi yerine AR ← PC kullanılması düşünülebilirdi ancak, veri yolunu aynı anda iki cihaz kullanamaz.

İvedi Mod için T4 Adımı

T4. IDEC00.ADRMD1: $DR_L \leftarrow M[AR], PC \leftarrow PC+1$



İvedi moda sahip ADD komutunun önceki adımlarında PC, 2 kere arttırılmıştır. Bu komut 3 byte'lık olduğundan, 1 kere daha arttırılarak PC'nin bir sonraki komutu göstermesi sağlanmıştır

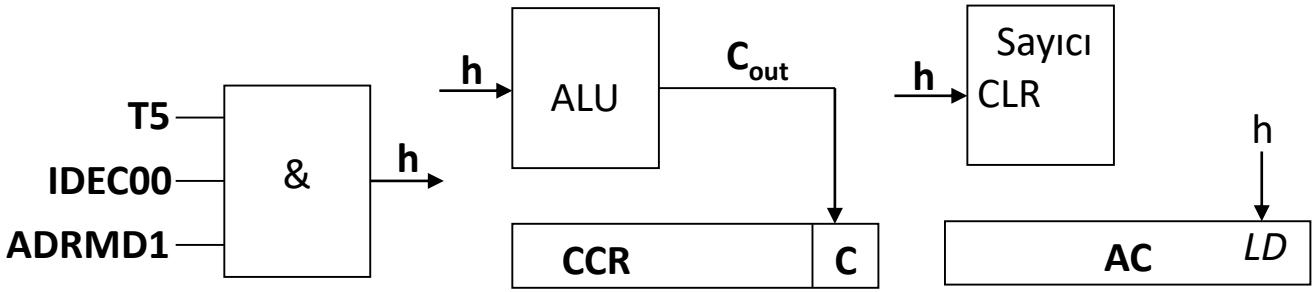


İvedi Mod için T5 Adımı

T5 adımında da ALU'da gerekli işlem yapılacak ve sayıcı sıfırlanacaktır:

T5. IDEC00.ADRMD1: AC ← AC+DR, C ← C_{out}, SC ← 0

T5 zamanlama diliminde ALU'da gerekli toplama işlemi yapılmış, işlemin sonucu AC'nin *LD* girişi aktive edilerek AC'ye aktarılmıştır. Toplama işlemi sonucunda elde oluşabileceğinden elde bayrağı güncellenmiştir. Ayrıca sayıcı da, bir sonraki komutun fetch işlemi için sıfırlanmıştır.



Örnek: Direkt adresleme moduna sahip ADD komutunun mikroişlem adımları.

ADD 1000h komutu gibi. Burada adres kısmının önünde bir işaret kullanılmamıştır, direkt adresleme modunu göstermektedir. 3 byte'lık bir komuttur.

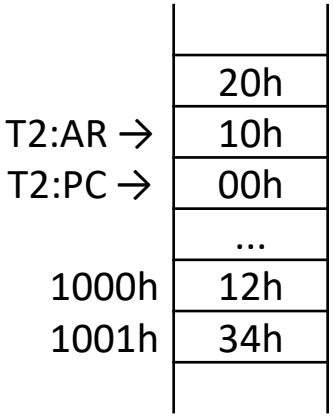
Direkt adresleme modunu kullanan ADD komutunun opcode'u $0\ 010\ 0000_2$ = 20h. 1000h adresindeki değerle akü toplanacak ve sonuç tekrar aküye aktarılacaktır.

Diğer komutlarda olduğu gibi bu komutun da T0, T1 ve T2 adımları aynıdır. Execute saykılı ise aşağıdaki gibidir.

KOMUTUN MİKRO İŞLEM ADIMLARI	
T3.IDEC00.ADRMD2:	$TR_H \leftarrow M[AR], AR \leftarrow AR+1$
T4.IDEC00.ADRMD2:	$TR_L \leftarrow M[AR], PC \leftarrow PC+1$
T5.IDEC00.ADRMD2:	$AR \leftarrow TR$
T6.IDEC00.ADRMD2:	$DR_H \leftarrow M[AR], AR \leftarrow AR+1$
T7.IDEC00.ADRMD2:	$DR_L \leftarrow M[AR]$
T8.IDEC00.ADRMD2:	$AC \leftarrow AC+DR, C \leftarrow C_{out}, SC \leftarrow 0$

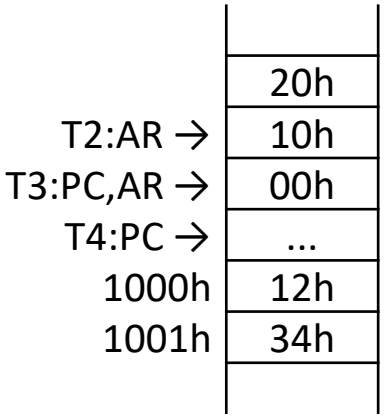
Direkt Mod için T3 ve T4 Adımları

T2 saykılının sonunda belleğin ve kaydedicilerin gösterdiği değerler:



T3. IDEC00.ADRMD2: $TR_H \leftarrow M[AR], AR \leftarrow AR+1$

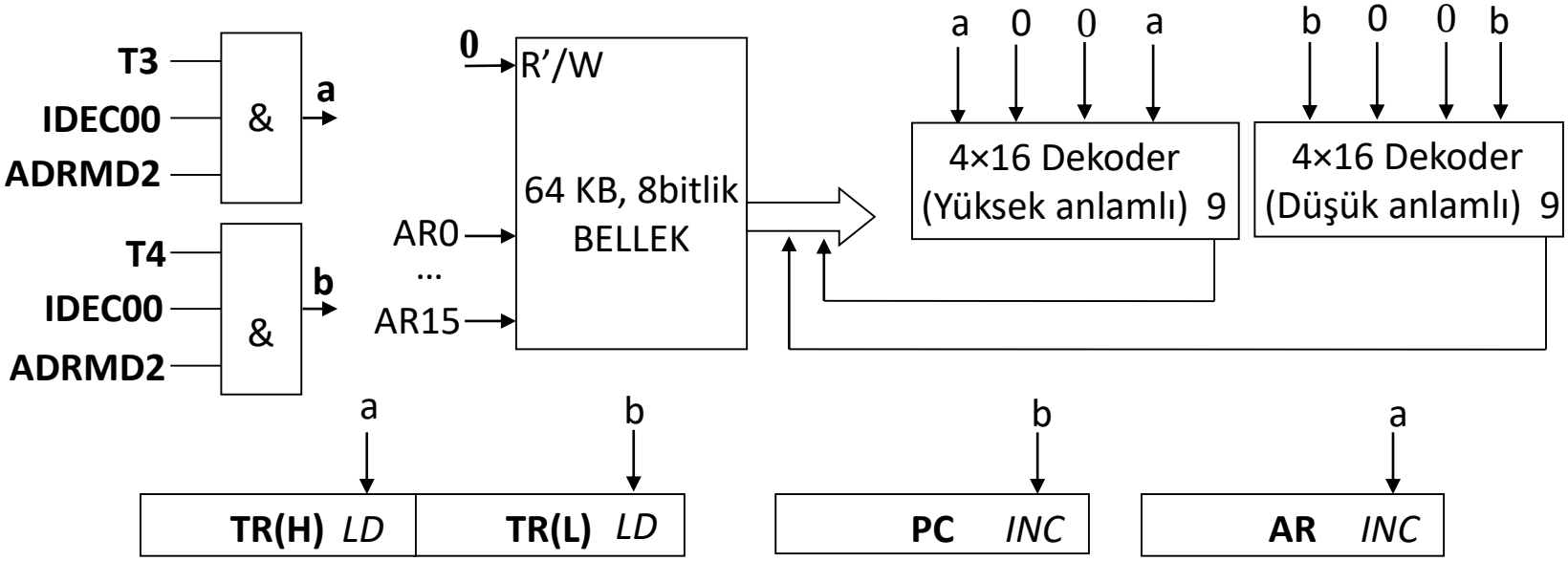
T4. IDEC00.ADRMD2: $TR_L \leftarrow M[AR], PC \leftarrow PC+1$



Direkt Mod için T3 ve T4 Adımları

T3. IDEC00.ADRMD2: $TR_H \leftarrow M[AR], AR \leftarrow AR+1$

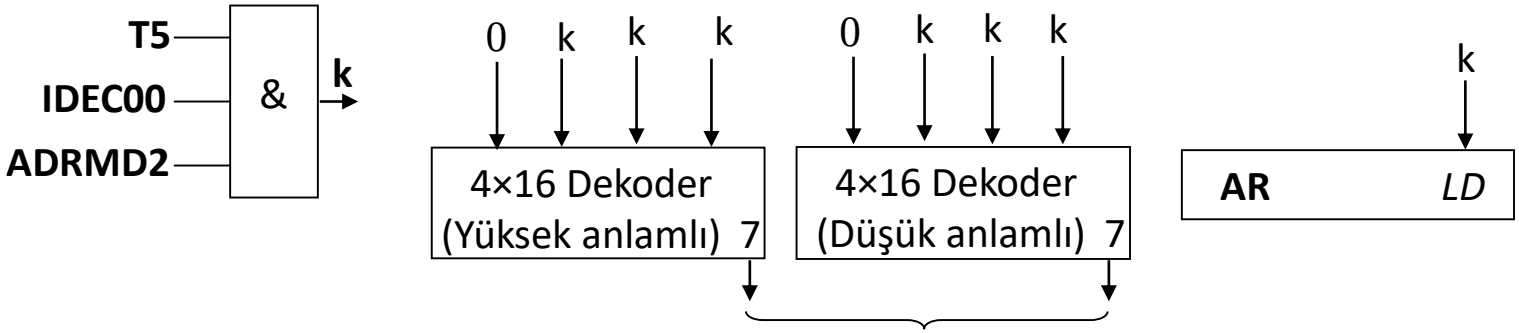
T4. IDEC00.ADRMD2: $TR_L \leftarrow M[AR], PC \leftarrow PC+1$



Direkt Mod için T5 Adımı

T5. IDEC00.ADRMD2: AR ← TR

(TR'nin gösterdiği yerdeki veriye erişmek için aktarım yapılmıştır)

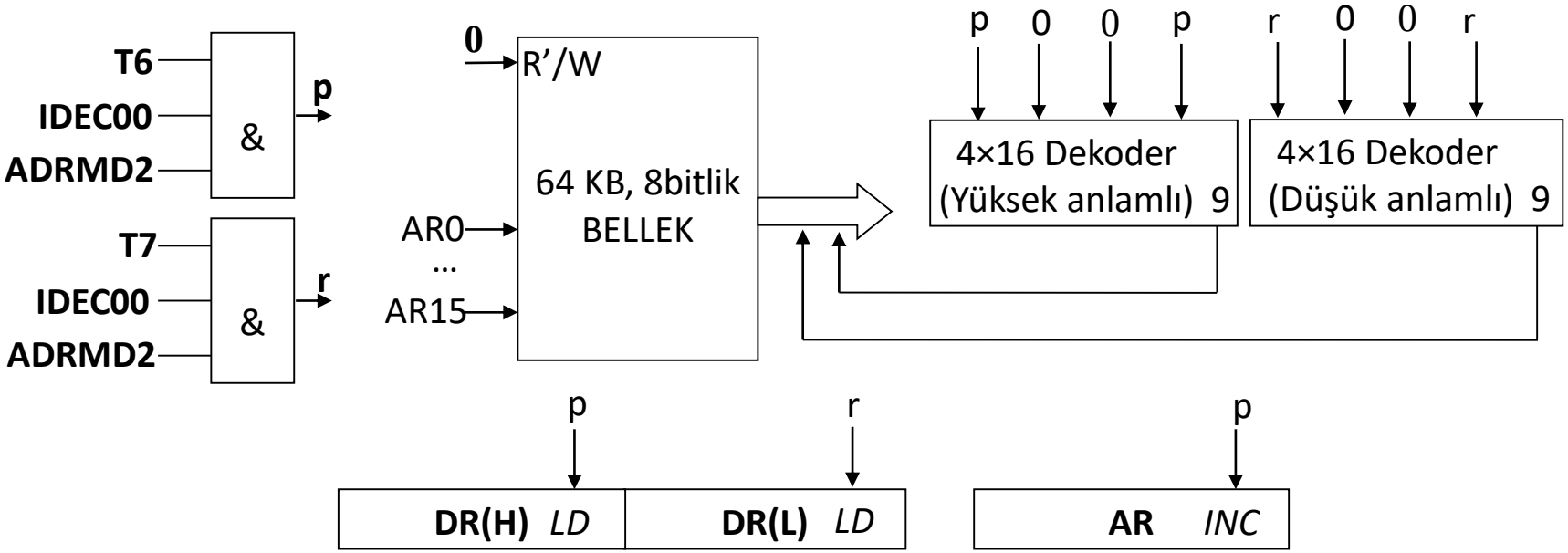


TR'nin içeriğini yola aktaran
tri-state'lere gider

Direkt Mod için T6 ve T7 Adımları

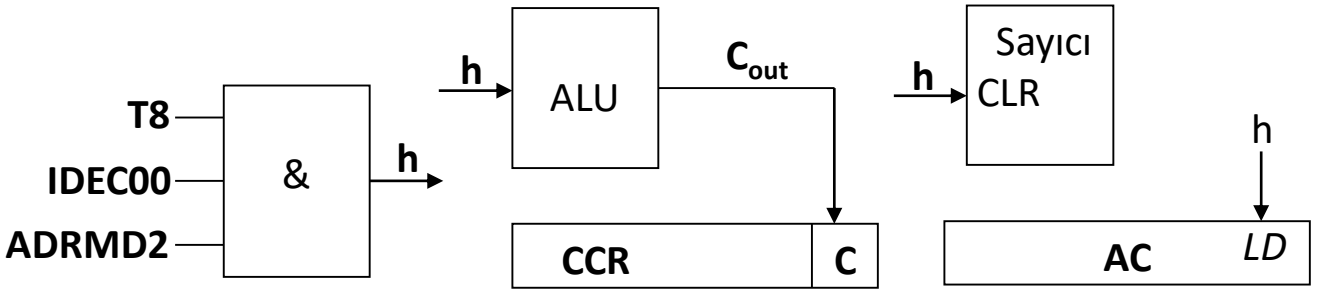
T6. IDEC00.ADRMD2 : $DR_H \leftarrow M[AR], AR \leftarrow AR+1$

T7. IDEC00.ADRMD2 : $DR_L \leftarrow M[AR]$



Direkt Mod için T8 Adımı

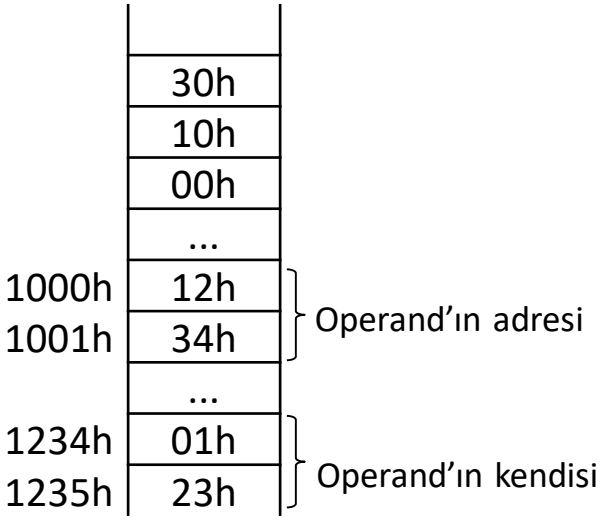
T8. IDEC00.ADRMD2: AC ← AC+DR, C ← C_{out} ,SC ← 0



Örnek: Dolaylı adresleme moduna sahip ADD komutunun mikroişlem adımları

ADD (1000h) komutu gibi. Burada adres kısmında parantezler kullanılmıştır, dolaylı adresleme modunu göstermektedir. 3 byte'lık bir komuttur.

Dolaylı adresleme modunu kullanan ADD komutunun opcode'u $0\ 011\ 0000_2$ = 30h. 1000h adresinde, operandın adresi bulunmaktadır. Belirtilen adresteki değerle (0123h) akü toplanacak ve sonuç tekrar aküye aktarılacaktır.



KOMUTUN MİKRO İŞLEM ADIMLARI	
T3. IDEC00.ADRMD3:	$TR_H \leftarrow M[AR], AR \leftarrow AR+1$
T4. IDEC00.ADRMD3:	$TR_L \leftarrow M[AR], PC \leftarrow PC+1$
T5. IDEC00.ADRMD3:	$AR \leftarrow TR$
T6. IDEC00.ADRMD3:	$TR_H \leftarrow M[AR], AR \leftarrow AR+1$
T7. IDEC00.ADRMD3:	$TR_L \leftarrow M[AR]$
T8. IDEC00.ADRMD3:	$AR \leftarrow TR$
T9.IDEC00.ADRMD3:	$DR_H \leftarrow M[AR], AR \leftarrow AR+1$
T10. IDEC00.ADRMD3:	$DR_L \leftarrow M[AR]$
T11. IDEC00.ADRMD3:	$AC \leftarrow AC+DR, C \leftarrow C_{out}, SC \leftarrow 0$