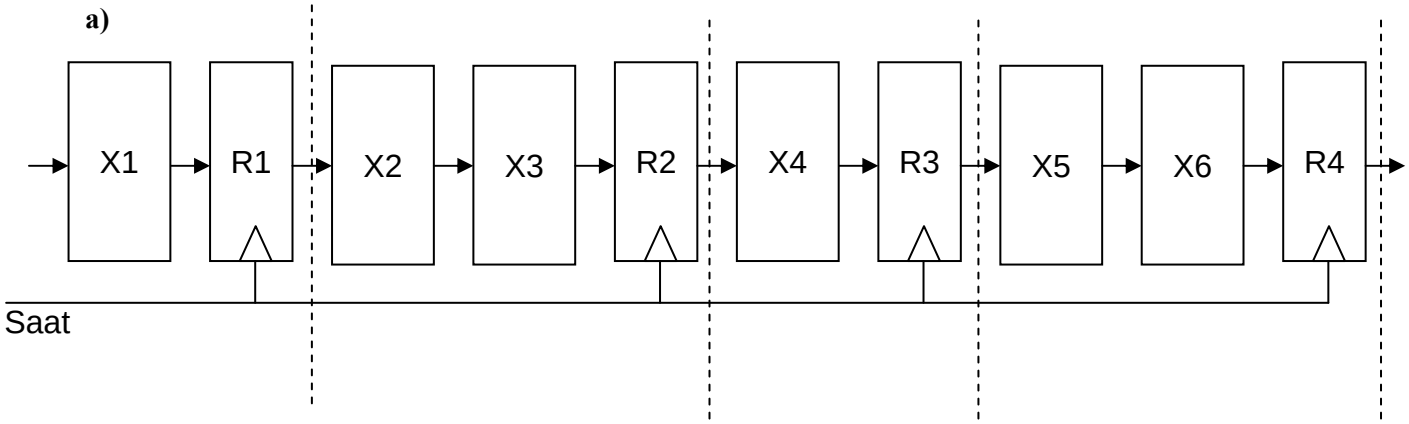
İ.T.Ü
Bilgisayar Mühendisliği Bölümü
Prof.Dr. Emre HARMANCI
Yrd.Doç.Dr. Feza BUZLUCA
Araş.Gör. Gülnur Selda KURUOĞLU

12.03.2009

BİLGİSAYAR MİMARİSİ 1. YILİÇİ SINAVI ÇÖZÜMLERİ

CEVAP 1:

a)



b) Yukarıdaki yapıda en yavaş segman 40+5 ns hızında olduğundan saat işareti tp=45 ns seçilecektir.
Birinci işlem segman sayısı * tp (4*tp) kadar sürecektir. 4*tp=4*45 =180 ns.
İkinci işlem (I_2) birinciden 1 saat darbesi (45 ns) sonra tamamlanacaktır. Toplamda 180+45 ns sürecektir.

c) İş hattı olmadan 10 işlemin süresi: t_n= 140 ns (Tüm X sürelerinin toplamı)
Hızlanma: s= n*tn/(k+n-1)tp = 10*140/(13*45) ≈ 2,4

d) İş hattı 3 segmanlı olarak da tasarlanabilir: (X1,X2), (X3,X4) , (X5,X6)
Bu durumda en yavaş segman 50 +5 =55 ns hızında olacaktır.
Bu yapı a şıkkındaki 4 segmanlı yapıya göre daha ucuzdur çünkü daha az sayıda saklayıcı içermektedir.
Birinci işin bekleme süresi daha kısadır: 3*55= 165 ns.
Ancak işlerin sayısı arttığında hızlanma oranı daha düşük olacaktır.

CEVAP 2:

a)

i)

```
*-----------------------------------------------------------
* Program    : k = AVG (i,j)
; Program sonlanınca $0508 adresinde (k) $ 00 00 00 0F değeri
; ( (i + j )/2 ) bulunur
*-----------------------------------------------------------
          ORG       $0500
i         DC.L      10
j         DC.L      20
k         DC.L      0

; ANA PROGRAM
          ORG       $1000
START:
                    MOVEA.W       #$4000, SP      ; Yigin $4000 adresinden baslatilmistir
                    PEA           i               ; i nin adesi yığına atılıyor
                    PEA           j               ; j nin adesi yığına atılıyor
                    PEA           k               ; k nın adesi yığına atılıyor ( k = (i + j) / 2 olacak )
                    JSR           AVG             ; AVG altprogramının cagirilmasi

                    MOVEA.W       #$4000,SP       ;yıgının bosaltilmasi
                    STOP          #$1000
; AVG altprogramı
          ORG       $2000
AVG
                    MOVEM.L   D0/A0,-(SP)         ; kullanılacak saklayıcılar yığına atıldı

                    MOVE.L    20(SP),A0           ; i nin adresi A0 da
                    MOVE.L    (A0),D0             ; D0 <- i

                    MOVE.L    16(SP),A0           ; j nin adresi A0 da
                    ADD.L     (A0),D0             ; D0 <- i + j
                    ASR.L     #1,D0               ; D0 <- ( i + j ) / 2

                    MOVE.L    12(SP),A0           ; k nın adresi A0 da
                    MOVE.L    D0, (A0)            ; sonuç k değişkenine yazıldı

                    MOVEM.L     (SP)+,D0/A0       ; Saklayicilarin eski degerleri yigindan çekiliyor
                    RTS

        END     START
```

```
J EQU $0504
K EQU $0508

ORG $0500

DCINT 10
DCINT 20
DCINT 0

ORG $0600

START:
        ADD     R0 , I, R10                             ; R10 <-- 0500 (i)
        ADD     R0 , J, R11                             ; R11 <-- 0504 (j)
        ADD     R0 , K, R12                             ; R12 <-- 0508 (k)
        CALL    AVG (R0), R13                           ; R13 <-- PC
                                                        ; PC  <-- AVG
                                                        ; CWP <-- CWP - 1
        NOP

ORG $0700
AVG:
        ; i adresi ana programda R10 = R26  (AVG de)
        ; j adresi ana programda R11 = R27  (AVG de)
        ; k adresi ana programda R12 = R28  (AVG de)
        ; donus adresi ana programda R13 = R29 (AVG de)

        LDL (R26)R0, R10                        ; R10 <- i
        LDL (R27)R0, R11                        ; R11 <- j
        ADD R11, R10, R10                       ; R10 <- i + j
        SRA R10, 1, R10                         ; R10 <- ( i + j ) / 2

        STL (R28)R0, R10                        ; k <- ( i + j ) / 2

        RET     (R29)0                          ; PC  <-- (R29) + 0
                                                ; CWP <-- CWP + 1
```

b) Komut alma çevrimleri dışındaki bellek erişimleri:

- **68000 de:**

Ana programda: parametrelerin yığına atılması: 4 adet bellek erişimi (dönüş adresi de yığına atılır)
Alt programda:

- kullanılan 2 saklayıcı yığına atılır: 2
- i, j, k adreslerinin yığından okunması: 3
- D0 <- i, D0 <- i + j işlemlerinde i ve j adreslerin içeriğinin okunması: 2
- M[k] <- (i + j) / 2 : sonusun k adresine yazılması: 1
- Yığından saklayıcıların eski değerlerinin çekilmesi: 2
- RTS komutu ile donus adresinin çekilmesi: 1
-

Toplam: 11 adet 32 bitlik erişim
MC 68000 işlemcisinin veri yolu 16 bit olduğundan 11 adet 32 bitlik erişim için aslında belleğe 22 adet 16 bitlik erişim yapılır.

- **RISC 1 de:**

Ana programda: saklayıcılara ivedi değerler yüklenir:0
Alt programda:

- R10 <- M[i] :1
- R11 <- M[j] :1
- M[k] <- (i + j) / 2 :1

Toplam: 3

SORU 3: (30 Puan)

a) Komut alma ile diğer bellek erişimlerinin çakışmaması için

- Komutlar ve veriler için ayrı bellekler kullanılabilir (Harward mimarisi). Berkler RISC 1'in yapısı von Neumann mimarisidir.
- Komutlar bellekten önceden okunarak bir komut kuyruğuna yerleştirilebilir.
- Eğer işlemleri hızlandıracak önlemler alınmazsa belleğe operand için erişim yapılırken iş hattına komut alınmaz o segman boş kalır.

b)

START: ADD R0,200,R18	I	A	E													
ADD R0,100,R17		I	A	E												
ADD R0,136,R16			I	A	E											
JMP BR,LOOP2(R0)				I	A	E										
LOOP1: SUB R16,4,R16					I	A	E									
LOOP2:LDL (R16)0,R21						I	A	E								
LDL (R17)0,R22							I	A	E							
ADD R21,R22,R23								I	A	E						
STL (R18)0,R23									I	A	E					
ADD R18,4,R18										I	A	E				
ADD R17,4,R17											I	A	E			
SUB R17,R16,R19												I	A	E		
JMP BNE,LOOP1(R0)													I	A	E	
ADD R17,4,R17														I	A	E

i)

```
START:    ADD R0,200,R18    ; İvedi veri 10 tabanında
          ADD R0,100,R17
          ADD R0,136,R16
          JMP BR,LOOP2(R0)  ;Her zaman dallan
          NOP
LOOP1:    SUB R16,4,R16
LOOP2:    LDL (R16)0,R21
          LDL (R17)0,R22
          NOP
          ADD R21,R22,R23
          STL (R18)0,R23
          ADD R18,4,R18
          ADD R17,4,R17
          SUB R17,R16,R19
          JMP BNE,LOOP1(R0)
          NOP
          ADD R17,4,R17
```

ii) Farklı şekillerde çözülebilir.

1. Çözüm:

```
START:    ADD R0,200,R18    ; İvedi veri 10 tabanında
          ADD R0,100,R17
          JMP BR,LOOP2(R0)  ;Her zaman dallan
          ADD R0,136,R16
LOOP1:    SUB R16,4,R16
LOOP2:    LDL (R16)0,R21
          LDL (R17)0,R22
```

```
                ADD R17,4,R17
                ADD R21,R22,R23
                STL (R18)0,R23
                SUB R17,R16,R19
                JMP BNE,LOOP1(R0)
                ADD R18,4,R18
                ADD R17,4,R17
```

2. Çözüm:

```
START:          ADD R0,200,R18      ; İvedi veri 10 tabanında
                ADD R0,100,R17
                ADD R0,136,R16
LOOP1:          LDL (R16)0,R21
                LDL (R17)0,R22
                ADD R17,4,R17
                ADD R21,R22,R23
                STL (R18)0,R23
                ADD R18,4,R18
                SUB R17,R16,R19
                JMP BNE,LOOP1(R0)
                SUB R16,4,R16
                ADD R17,4,R17
```