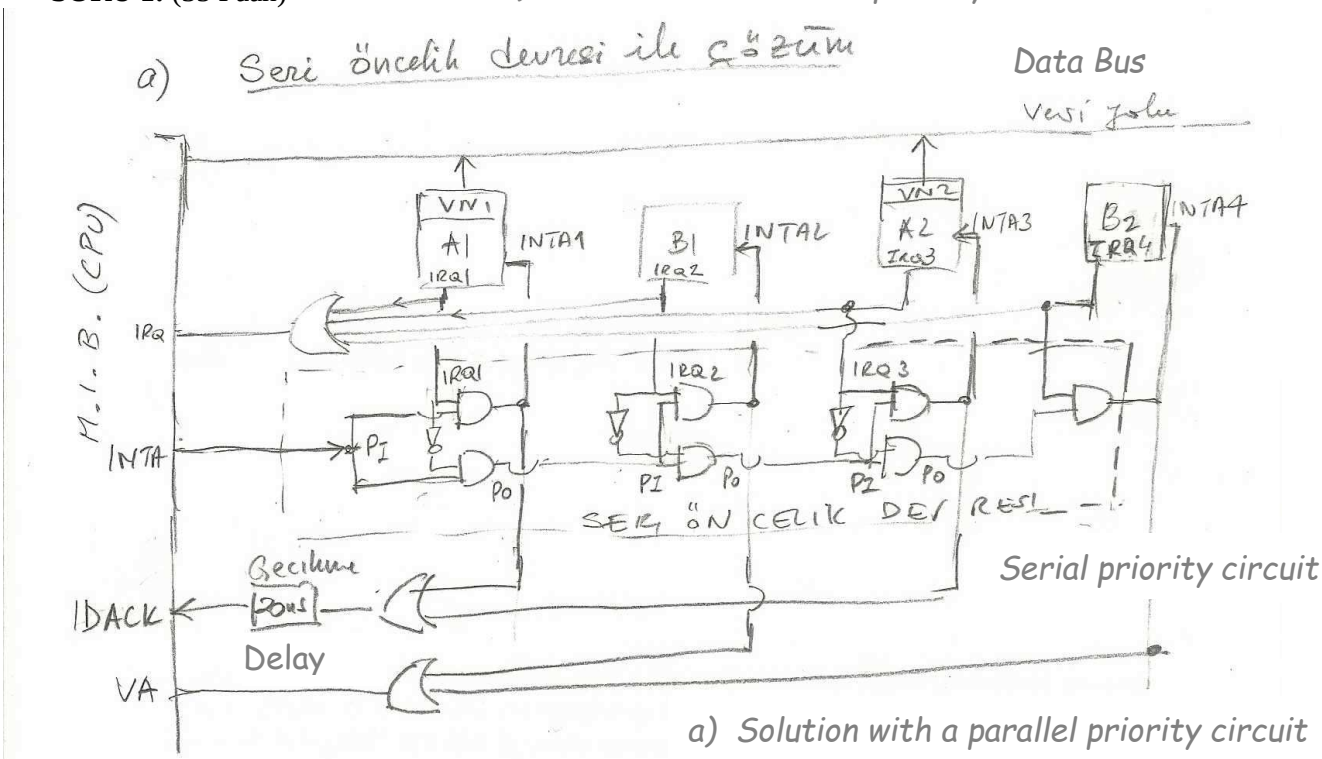
24.04.2014

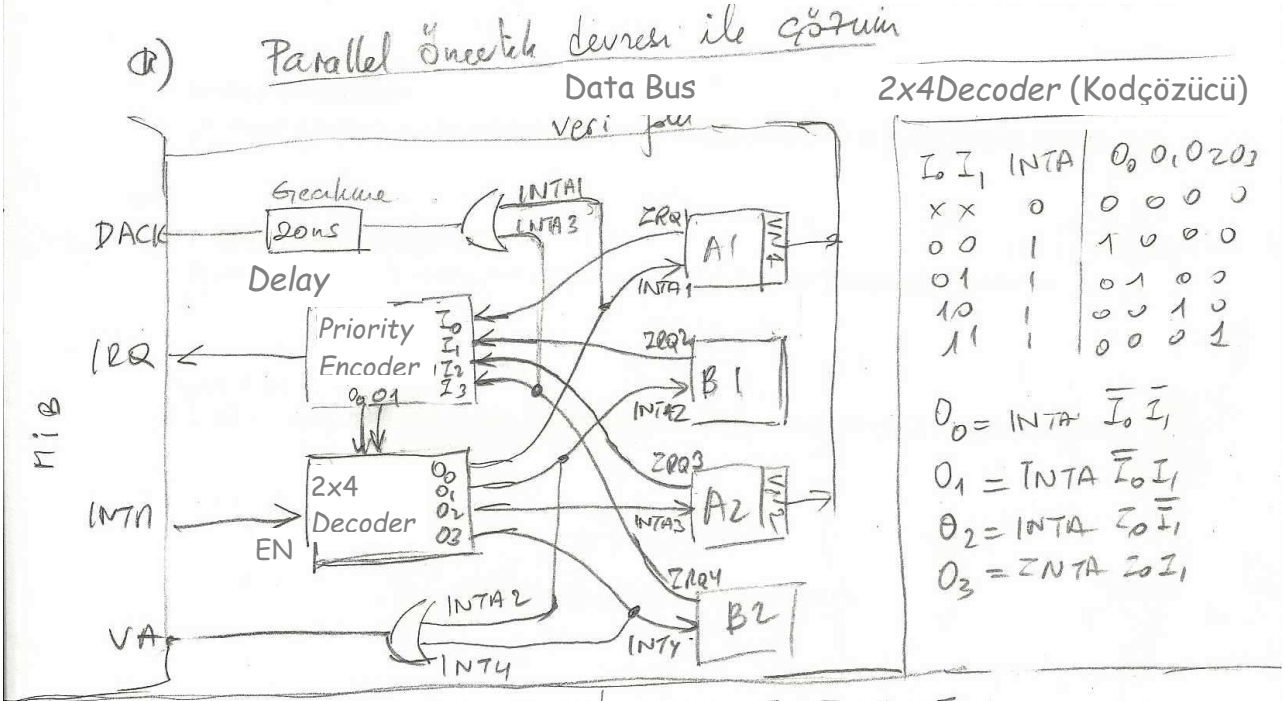
COMPUTER ARCHITECTURE 2nd MIDTERM SOLUTIONS
BİLGİSAYAR MİMARİSİ 2nci YILİÇİ SINAV ÇÖZÜMLERİ

QUESTION 1: *(35 Points)*
SORU 1: (35 Puan)

a) Solution with a serial priority circuit



a) Solution with a parallel priority circuit



Öncelik Kodlayıcı
Priority Encoder

I0	I1	I2	I3	IRQ	O0	O1
0	0	0	0	0	×	×
1	×	×	×	1	0	0
0	1	×	×	1	0	1
0	0	1	×	1	1	0
0	0	0	1	1	1	1

IRQ = I0 + I1 + I2 + I3
O0 = Ī0 Ī1
O1 = Ī0 I1 + Ī0 Ī2

b) İşaretler

b) *Signals*

IRQ → MIB (CPU)

(CPU) MIB → INTA (IRQ1 is removed), VN1 → Veri yolu (Data Bus)

INTA1 → A1 (IRQ1 etkisizleşir)

DACK → MIB (20ns gecikme)

VN → Veri yolu (20ns gecikme) *To Data Bus (after 20ns delay)*

(MIB DI (disable interrupt) yapar)
(MIB A1 ilgili ISR'yi yürütür)
(MIB IE (enable interrupt) yapar)
} İç işler (*Internal operations*)

(Kesilen uygulamada bir komut yürütülür) (*One instruction of the interrupted program is executed*)

IRQ → MIB

MIB → INTA

INTA2 → B1 (IRQ2 etkisizleşir) (*IRQ2 is removed*)

INTA2 → VA

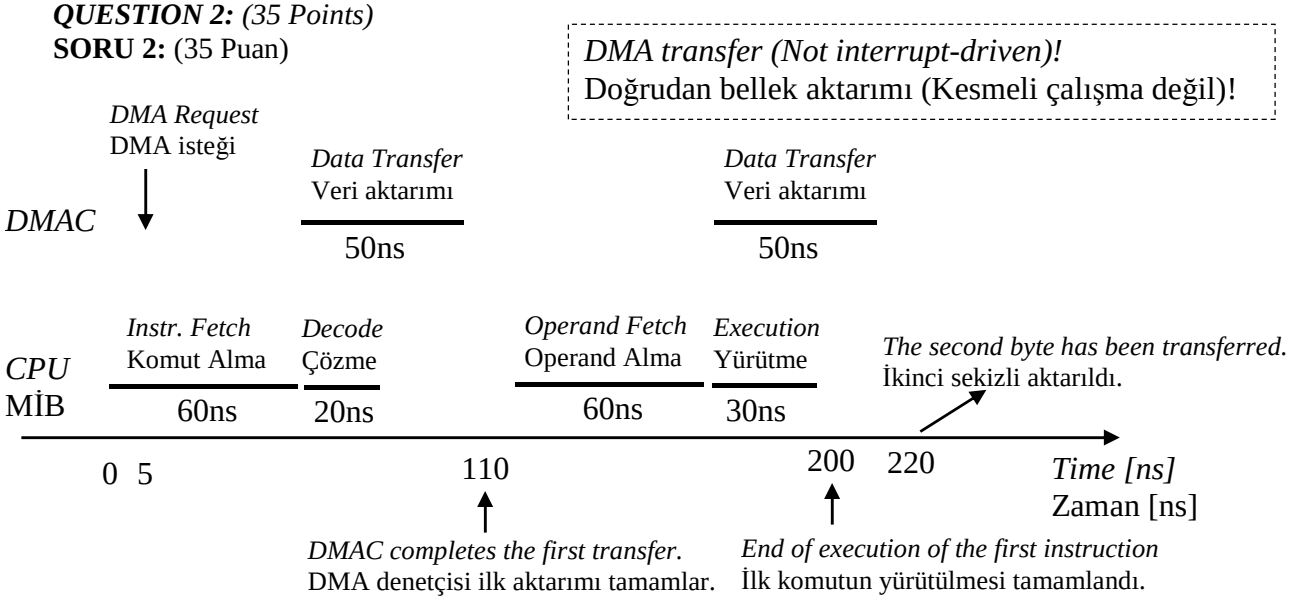
(MIB DI)
(ISR "otovektör" yoklama ile B1'in kesme gönderdiğini keşfeder) (*In the ISR using the software polling the source of the interrupt (B1) is determined.*)
(ISR(B1) yürütülür)
(Kesilen uygulamaya dönülür) (*Return to the interrupted program.*)

c) A1 ve A2 için VN1 veya VN2 ile vektör tablosundan B1 ve B2 için dallanılan ISR "otovektör" yoklama ile önce B1'in ilgili bayrağını, bayrak etkin değilse sonra B'nin ilgili bayrağını test eder. Buna göre ilgili ISR (sürücü programa) dallanır.

c) For Type A: The vector number is given by the device. The CPU gets the start address of the ISR from the interrupt vector table using the vector number (VN1 or VN2).

For Type B: In the ISR of the "autovector" using the software polling flags of the devices are checked (first B1 then B2). According to these flags the program (ISR of autovector) jumps the driver program of the B1 or B2.

QUESTION 2: *(35 Points)*
SORU 2: (35 Puan)



a) (*5 points* / 5 puan)
*CPU completes the current bus cycle (Instruction fetch) and isolates itself from the system bus. The DMAC transfers the first data. Since the type of the DMAC is **fly-by** (Implicit) data is transferred in 50 ns.*
Time= 60 + 50 = 110ns
MİB o andaki yol çevrimini tamamlar (komut alma) ve kendini sistem yolundan yalıtır. DMA denetçisi ilk sekizliyi aktarır. DMA denetçisi örtülü olarak çalıştığı için veri 50ns'de aktarılır.
Zaman= 60 + 50 = 110ns

b) (*5 points* / 5 puan)
*Instruction decoding and execution cycles of the CPU can run in parallel with DMA transfers. Since the DMAC uses the **cycle-stealing** technique, after the transfer of the first byte it will give the bus to CPU. After the operand fetch the CPU executes the instruction.*
Time= 60 + 50 + 60 + 30 = 200ns
MİB'in komut çözme ve operand alma çevrimleri DMA aktarımları ile paralel yürüyebilir. DMA denetçisi çevrim çalma yöntemiyle çalıştığı için ilk sekizliyi aktardıktan sonra yolu MİB'e verir. MİB operandı aldıktan sonra komutu yürütür.
Zaman= 60 + 50 + 60 + 30 = 200ns

c) (*10 points* / 10 puan)
*During one instruction cycle the DMAC transfers **two bytes in 220 ns**.*
10 bytes are transferred in 5 x 220 = 1100 ns.
*Time= 5 * 220 = 1100ns*
DMA denetçisi bir komut çevriminde **iki sekizliyi 220 ns'de** aktarmaktadır.
10 sekzili 5 x 220 =1100ns'de aktarılır.
Zaman= 5 * 220 = 1100ns

Consider the difference.
Farka dikkat ediniz.

After the transfer of the 10 bytes the CPU runs each instruction in 170 ns.
*Time= 1100 + 5*170 = 1950ns*
On sekizli aktarıldıktan sonra MİB her komutu 170ns'de yürütür.
Zaman= 1100 + 5*170 = 1950ns

d) (*5 points* / 5 puan)
Remember; interrupt requests are checked after the execution of the instruction. If there is a request the CPU enters the interrupt cycle (given in the question). The data is transferred in the ISR.
Time= 60 + 20 + 60 + 30 + 200 + 500 = 870ns
Hatırlatama; kesme istekleri komut yürütüldükten sonra yoklanır. Eğer kesme varsa MİB kesme çevrimine girer (soruda verilmişti). Veri aktarımı kesme hizmet programının içinde yapılır.
Zaman= 60 + 20 + 60 + 30 + 200 + 500 = 870ns

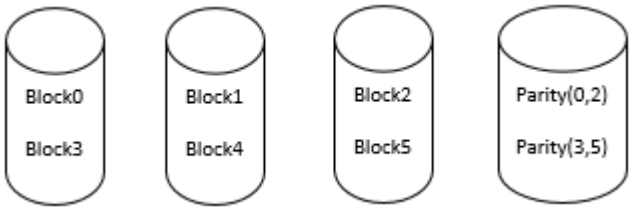
d) (*10 points* / 10 puan)
After the execution of each instruction one byte is transferred in the ISR in 870ns.
*Time= 10 * 870 = 8700ns*
Her komutun yürütülmesinden sonra kesme hizmet programında bir sekizli toplam 870 ns'de aktarılır.
Zaman= 10 * 870 = 8700ns

The execution of the last instruction is completed before the transfer of the last byte. Therefore we subtract the durations of the housekeeping and ISR (200ns+500ns) from total time (8700ns).
Time= 8700 - 700 = 8000ns
Son komutun yürütülmesi son sekizlinin aktarımından önce tamamlanır. Bu nedenle kesme hazırlık işlemleri ve kesme hizmet programı sürelerini (200ns + 500ns) toplam süreden (8700ns) çıkartıyoruz.
Zaman= 8700 - 700 = 8000ns

Note: *Compare the results obtained by the DMA technique and interrupt-driven technique. The advantage of the DMA technique is obvious.*
Not: DMA yöntemi ve kesmeli çalışma ile elde edilen sonuçları karşılaştırınız. Doğrudan bellek erişimi yönteminin avantajı açıkça görülmektedir.

QUESTION 3: *(30 Points)*
SORU 3: (30 Puan)

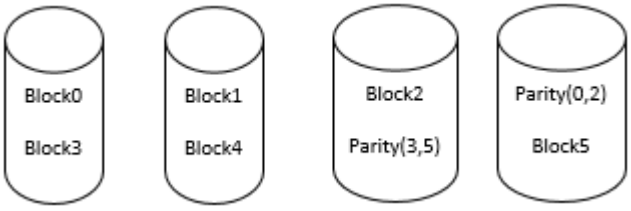
a)



*i) Two different blocks from two different disks can be read in **ta.***
İki farklı diskteki iki blok paralel olarak **ta** sürede okunur.

*ii) Two read and two write operations (**2ta**) should be performed for an update operation in RAID 4 (See the lecture notes). Since parity update operations cannot be performed independently (in parallel) (there's only one parity disk), it takes **4ta** to update words of two blocks in two different disks.*
RAID 4'te her yazma işlemi iki diski okumayı iki diske yazmayı gerektirir (Ders notlarına bakınız). Okumalar ve yazmalar kendi aralarında paralel olduğu için bir diske yazma **2ta** sürer. Ancak tek bir eşlik diski olduğundan eşlik güncelleme işlemleri paralel olarak yapılamaz. İki farklı dikteki iki bloğa yazma işi **4ta** sürer.

b)



*i) Same as in RAID 4: **ta**.*
RAID 4 ile aynıdır: **ta**.

*ii) For each data update two read and two write operations are necessary. Different from RAID4, now parity update operations can be performed in parallel, because parity strips are distributed to different disks: **2ta**.*
Her veri güncellemesi için iki okuma ve iki yazma işlemi gereklidir. RAID 4'ten farklı olarak eşlik güncelleme işlemleri paralel yapılabilir, çünkü eşlik bilgileri farklı disklere dağıtılmıştır: **2ta**.