

1	2	3	4	Toplam
1	1	1	3,4	PÖÇ

1. a. Which addressing mode should be used to make a program relocatable so that it can run in any address range under a multitasking operating system. $\Leftrightarrow$ Bir programı, çok görevli bir işletim sistemi altında hangi adres aralığına yüklenirse yüklenin çalıştırılabilir, yani yeniden-yerleştirilebilir (relocatable) yapmak için hangi adresleme modu kullanılmalıdır.

Göreceli (Relative) adresleme modu kullanılmamıştır.

- b. How is the effective address computed in this addressing mode. $\Leftrightarrow$ Bu adresleme modunun efektif adresi nasıl hesaplanmaktadır.

Efektif Adres EA = PC + ofset

2. a. The output of Enable Interrupt Flip-Flop (EIF) which can be set (Enable Interrupts, EI instruction) or reset (Disable Interrupts, DI instruction) by the user is AND'ed with microprocessors Interrupt ReQuest (IRQ) input to mask interrupt requests sometimes. Explain the reason for masking IRQ input by masking with EIF in one sentence. $\Leftrightarrow$ Kullanıcı tarafından kurulabilen (EI emri ile kesmeleri yetkilendirir) veya sıfırlanabilen (DI emri ile kesmeleri yetkisiz kılar) Kesme Yetkilendirme Flip-Flop'u EIF çıkışı, işlemcinin Kesme İsteği (IRQ) girişi ile VE işlemine sokularak bazen kesme isteklerinin yetkisiz kılınması için kullanılmaktadır. IRQ kesme istek girişinin neden bazen EIF ile maskelenerek yetkisiz kılındığını **bir cümle ile** açıklayınız.

Programların (veya işletim sistemi süreçlerinin) sınırlı sürede bitirmesi gereken iş(ler)i, bu işlemler yapılırken araya hiçbir şekilde başka işlem sokmadan yürütmek ve tamamlayabilmek için kesmeler bu işlemlere başlanmadan önce yetkisiz kılınır ve işlem(ler)in tamamlanmasından sonra tekrar yetkilendirilir.

b. Microprocessor disables interrupts by resetting EIF after saving the PC to stack and before ACKnowledge the Interrupt requesting device with IACK if interrupts are enabled and a valid Interrupt ReQuest IRQ signal is detected at the end of an instruction execution. Explain the reason to temporarily disable interrupts by resetting EIF in one sentence. $\Leftrightarrow$ Kesmeler yetkilendirmiş ve çalıştırılan bir emir sonunda geçerli bir kesme giriş işareti IRQ alınmışsa, işlemci PC'yi yığına sakladıktan sonra ve kesme isteği üreten aygıtı kesmeyi kabul ettiğini gösteren IACK işaretini göndermeden önce, EIF'yi sıfırlayarak kesmeleri yetkisiz kılar. Geçici olarak kesmelerin neden yetkisiz kılındığını **bir cümle ile** açıklayınız.

Kesme servis altprogramına gitmeden önce EIF'yi sıfırlayarak kesme girişlerinin yetkisiz kılınmasının amacı, kesme servis programına gidildiğinde henüz kesme isteğini kaldırmamış olan kesme üreten cihazın yeniden kesmeye neden olmaması içindir.

- c. Under which circumstances the interrupt service routine should enable interrupts again by an EI instruction before servicing the interrupt requesting device? $\Leftrightarrow$ Hangi koşullarda kesme servis programı kesme üreten aygıtı hizmet vermeden önce kesmeleri tekrar bir EI emri ile yetkilendirmelidir.

Kesme üreten cihaz, eğer en yüksek öncelikli ve zaman sınırlı bir iş yürütmüyorsa, kendisinden daha yüksek öncelikli ve kritik sürede servis yürütecek cihazların kesme yapabilmesi için kendiservis programına başlamadan önce kesmeleri tekrar yetkilendirmelidir.

3. a. What is the most important problem that needs to be addressed while interfacing a very fast processor with an moderately fast I/O device such as a printer. $\Leftrightarrow$ Çok hızlı bir işlemci ile örneğin yazıcı gibi ortalama hızlı bir G/Ç cihazını birbirine bağlarken çözülmesi gereken en önemli problem nedir.

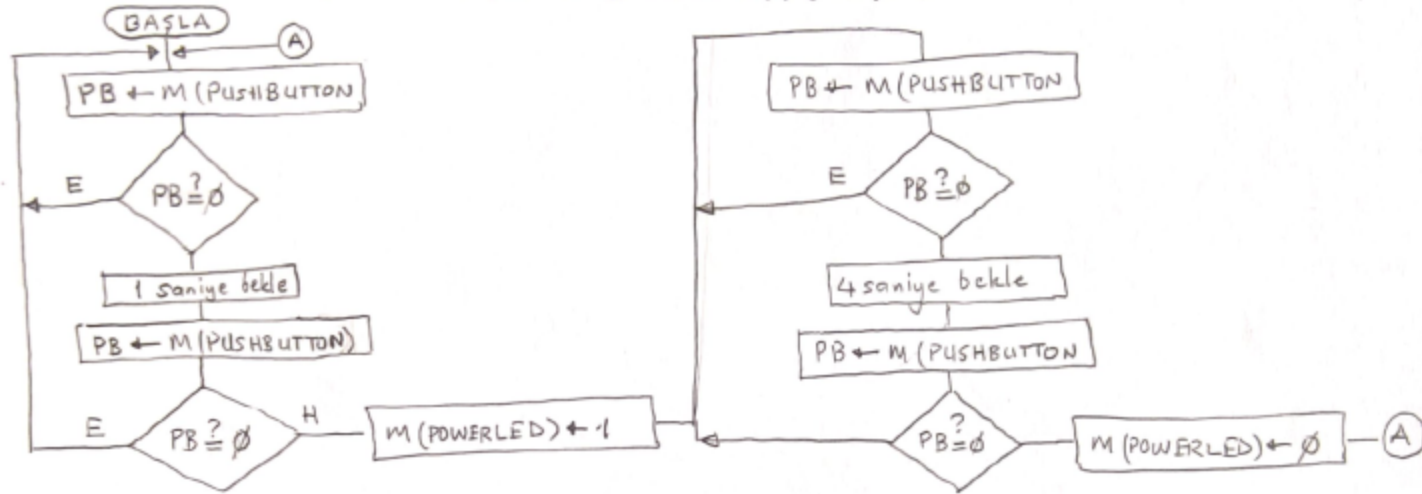
Hızları arasında çok büyük farklılık olan iki birimin sağlıklı haberleşmesi için yavaş olanın hızlandırılma olasılığı bulunmadığından, hızlı olan yavaşlatılmalıdır. Bu nedenle hızlı cihazın G/Ç hızının yavaş cihazınkinden düşürülmesi en büyük sorundur.

- b. How is the problem mentioned in (a) handled successfully by both modern Operating Systems and I/O devices. $\Leftrightarrow$ (a)'da söz edilen problemin modern İşletim Sistemleri ve G/Ç aygıtları tarafından nasıl ele alındığını bir cümle ile açıklayınız.

Bu sorunu çözmenin en etkili yolu, hızlı olan cihazın hızında iletişim kurmayı sağlayan tampon (buffer) veya kuyruk (queue) gibi donanımlar kullanarak, hızlı cihazı yavaşlatmadan iki cihaz arasında G/Ç işlemlerini yürütmektir. (örneğin yazıcı ve manyetik disklerdeki donanım ile sağlanan tampon bellek)

4. Power (on-off) switch of battery operated electronic devices are simple toggle switch activated by push-button with an advanced timing mechanism to circumvent inadvertant toggling of power states. Design a microprocessor based power switch activated by a push button to power "on" and "off" a microprocessor controlled system. You may assume that push button data which you can input (load) from address PUSHBUTTON is hardware De-Bounced by an S-R latch. Assuming the device is "off" initially, your algorithm should turn the system on if the push-button output is active for more than 1 second. The algorithm should indicate that the power is "on" by turning a LED on at symbolic address POWERLED. Once the device is active, the same push-button should be pressed for more than 4 seconds to power "off" in order not to shut-down the system by accidentally pressing the power button momentarily. The algorithm should indicate that the power is "off" by turning off the LED. State all other assumptions you have to make in order to design the power switch with timing. $\Leftrightarrow$ Batarya ile çalıştırılan elektronik cihazların güç (açma-kapama) anahtarı, güç durumunu istemeden değiştirmeyi engellemek üzere gelişmiş bir zamanlama mekanizması olan ve basmalı tuşla çalıştırılan basit bir devimeli anahtardır. Mikroişlemci denetimli bir sistemi "açmak" ve "kapatmak" için mikroişlemcili güç tuşu tasarlayınız. PUSHBUTTON adresinden giriş (yükleme) yapabileceğiniz basma tuşu işareti sıçramasının donanım düzeyinde bir S-R kilit ile giderilmiş olduğunu varsayabilirsiniz. Cihazın başlangıçta kapalı olduğu durumda bulunduğu varsayımı ile algoritmanız eğer düğme çıkışı 1 saniyeden daha uzun süre aktifse cihazı açmalıdır. Algoritmanız sistemin "açık" olduğunu gösteren POWERLED sembolik adresindeki LED'i yakmalıdır. Cihaz açık iken, güç düğmesine yanlışlıkla çok kısa süreli basmalarda sistemin kapatılmasına engel olmak için, aynı güç tuşuna en az 4 saniye boyunca basarak kapatmak gerekmektedir. Algoritma, sistemin kapatılmış olduğunu göstermek için LED'i de söndürmelidir. Zamanlamalı güç düğmesini tasarlarken yapmak zorunda kaldığınız diğer tüm varsayımları belirtmeniz gerekmektedir.

- a. Draw the flowchart of the power switch algorithm. $\Leftrightarrow$ Güç anahtarı algoritmasının akış çizgesini çiziniz.



- b. Code power switch algorithm in any high level language. $\Leftrightarrow$ Güç anahtarı algoritmasını herhangi bir yüksek seviyeli dil ile kodlayınız.

```
while (true) {
    döngü1: do {
        PB = inport(PUSHBUTTON)
    } while (PB == 0)
    delay(1000) // 1000 ms bekle
    PB = inport(PUSHBUTTON)
    if (PB == 0) goto döngü1
    outport(POWERLED, 1)
}
```

```
döngü2: do {
    PB = inport(PUSHBUTTON)
} while (PB == 0)
delay(4000) // 4000 ms bekle
PB = inport(PUSHBUTTON)
if (PB == 0) goto döngü2
outport(POWERLED, 0)
goto döngü1
```

- c. Code power switch algorithm in Assembly language. $\Leftrightarrow$ Güç anahtarı algoritmasını Assembly dili ile kodlayınız.

```
döngü1: in A, PUSHBUTTON
        and A, A
        jz döngü1
        mov A, #63
        call delay
        in A, PUSHBUTTON
        and A, A
        jz döngü1
        mov A, #FF
        out POWERLED, A
```

```
döngü2: in A, PUSHBUTTON
        and A, A
        jz döngü2
        mov A, #255
        call delay
        in A, PUSHBUTTON
        and A, A
        jz döngü2
        mov A, #0
        out POWERLED, A
        jz döngü1
```