1. **Határozza mez az I/O csatornákat vezérlő speciális funckiójú regiszterek szerepét. Melyik mit állít be? Ne feledkezzen meg az I/O csatorna esetleges analóg funkciójáról sem**

* a mikrovezérlők egyik fő erősségét az határozza meg, hogy menynire sokoldalúan lehet használni a kivezetéseit
* a mikrovezérlők esetén megkülönböztetünk 4 típusú regisztert
  + **Irány meghatározó regiszter** **(TRISx)**, ha a megfelelő bitje 1-es akkor bemenet, másként kimenet
  + **Kimeneti reteszelő regiszter (LATx)**, az adott álapotnak egy bizonyos feszültségszint felel meg, 1-nek 5V, 0-nak 0V
  + **Bemeneti reteszelő regiszter (PORTx)**, a bemenetek állapotát rögzíti a beolvasás művelet végrehajtásának pillanatában
  + **Speciális funkciókat kezelő regiszter**, ez kezelheti az ADDA-t, külső megszakításokat...
* **Analog funkcio?**

1. **Magyarázza el röviden a hardveres időzítés két módszerét és határozza meg a hardveres időzítés módszer fő előnyét a szoftveres időzítéssel szemben**

* hardveres időzítésnek nevezzük azt a független időmérési módszert, mely során a szükséges időintervallumot egy időzítő áramkör segítségével mérjük ki
* az időtartam leteltét egy jelzőbit beállítása jelez, mely okozhat megszakítást, vagy felhasználóra van bírva az észlelése
* működésük egyszerű: az áramkör a központi egységtől függetlenül addig számlálja a bemenetre kapcsolt órajel impulzusokat, amíg el nem éri a túlcsorduláshoz vezető impulzusszámot, ilyenkor pedig beállít egy jelzőbitet
* felépítéstől függően az áramkör számolhat előre vagy visszafele
* lehetőség van egy időzítőt 0-tól különböző értékre inicializálni, ekkor elérhető a teljes számlálási periódusnál kissebb időintervallum kimérése
* a **valós hardveres** időzítés a timerre alapul, az időtartam leteltét az időzítő áramkör egy megszakításkéréssel jelzi
* pontos módszer, mivel nem szakítható meg, elérhetővé teszi időosztásos szoftveres végrehajtási szálak ütemezését is
* a **kombinált módszer** is hardveres időzítés, habár nem annyira pontos
* a timer ugyanúgy számol, beállítja a flag-et, viszont nem kér megszakítást, az idő leteltét a felhasználó kell ellenőrizze, ezért csak azt mérhetjük ki, hogy letelt-e már az idő, azt nem hogy pontosan mikor
* a hardvers időzítéssel ellentétben a szoftveres nem független, az utasításvégrehajtás idejére alapszik, ha ismerjük ezt, egy ciklusban addig pörgetjük, míg eltelik a megfelelő idő
* nagy hátránya, hogy teljességében lefoglalja a proceszort és sokkal pontatlanabb is, mivel megszakítás is félbeszakíthatja

1. **Magyarázza el hogyan működik a „Capture” áramkörrel ellátott időzítő. Adjon egy példát a felhasználására**

* Capture = Számláló-mintavételező áramkör
* ez egy olyan számláló áramkör, mely egy külső jel előre meghatározott élére automatikusan és azonnal elmenti egy időzítő pillanatnyi értékét, ez a **Capture**
* a mérés lényegében a periódust méri két él között
* a mérési pontosság megegyezik az adott számláló órajel legkissebb órajelének pontosságával
* ha viszont túl alacsony periódusokat mérünk, nem lehet értelmezni az eredményt, ezért létezik előosztó is
* ha túl nagy frekvenciát szeretnénk mérni, ki kell egészítsük a szoftvert egy szoftveres számlálóval, mely a timer túlcsordulását számolja
* Pelda:
  + Frekvencia mérés = 1/periodus

1. **Magyarázza el a megszakítás kérés kiszolgálásának menetét érintve a következő fontos részleteket: mi történik a programszámláló értékével, mi történik a megszakítás vektor címértékével, milyen adatokat kell elmenteni a megszakítás rutin elején, milyen jelzőbiteket kell ellenőrizni a kiszolgáló rutin elején, mi történik a megszakítás rutin végén**

* a megszakítással lehetővé válik olyan események figyelése, melyet a főprogram futása közben lehetetlen ciklikusan figyelni
* mivel az esemény bekövetkeztét a főporgram futtatási lépéseire nézve kiszámíthatatlannak tekintjük, ezért kiszolgálásukkor bizonyos szabályokat be kell tartani
* a megszakítás menete a következő lépésekkel írható le:
  + a külső folyamat beállít egy jelzőbitet, melyet a processzor minden utasítás után ellenőriz, ha van megszakítás, akkor elkezdi ennek kezelését
  + először elmenti a programszámláló következő utasítását a verembe **(PC+1)**
  + melléje kerülnek az állapotjelző bitek, az ALU munkaregiszterei, általános munkaregiszterek és egyéb olyan regiszterek, melyek szükségesek a főprogram futásához
  + a programszámláló felülíródik a megfelelő megszakításvektor címével; ezen a címen általában egy JUMP utasítás található, mely a programszámlálót ráállítja a kiszolgáló függvényre
  + a kiszolgáló rutin elején ellenőriznünk kell a megszakítás engedélyező bitet, illetve ha csak egyetlen megszakítás vektor létezik, akkor a megszakítást jelző biteket is, ha mindkét bit aktív, elkezdődhet a rutin végrehajtása
  + a kezelőrutin végrehajtása után az elmentett fontos állapotregiszterek, akkumulátorok, munkaregiszterek és más változók visszaállítására kerül sor
  + utolsó lépésként az elmenttt programszámláló értékét vissza kell állítani

1. **Irja le röviden a mikrovezérlőbe beépített Watchdog speciális áramkör szerepét és működésének lényegét**

* a watchdog egy időzítő, melyet hibás működés kezelésére használnak leggyakrabban
* normális működés során, a mikroprocesszor folyton alapállásba helyezi az időzítőt, így megakadályozza, hogy túlcsorduljon
* ha hardver vagy szoftver hiba miatt a watchdog nem kerül alapállásba, eléri a túlcsordulást, mely egy jelzést küld
* ezen jelzés hatására hibajavító lépések történnek, általában a számítógép alaphelyzetbe állítása történik
* ez azzal jár, hogy a program futása előről indul
* a módszer nem garantálja a program helyes működését, csak biztonsági kapcsolóként van jelen