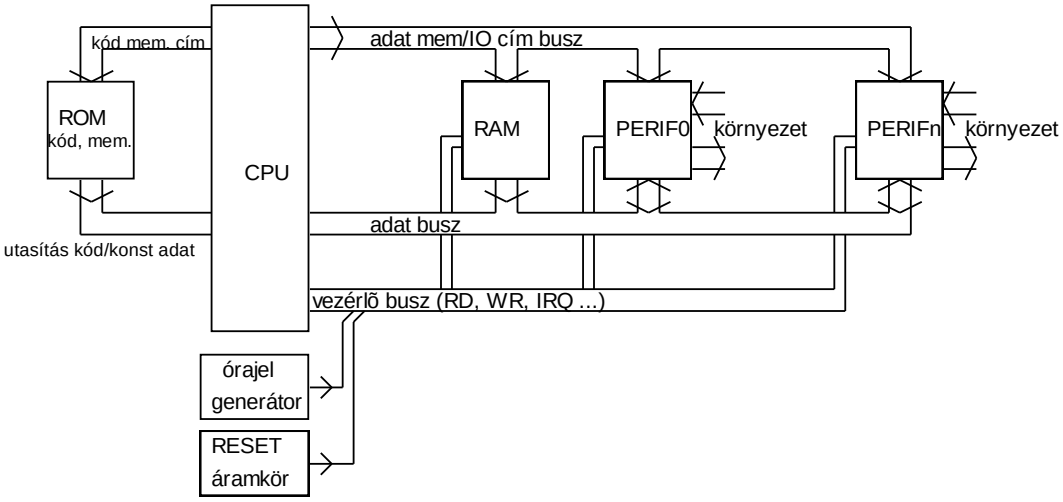
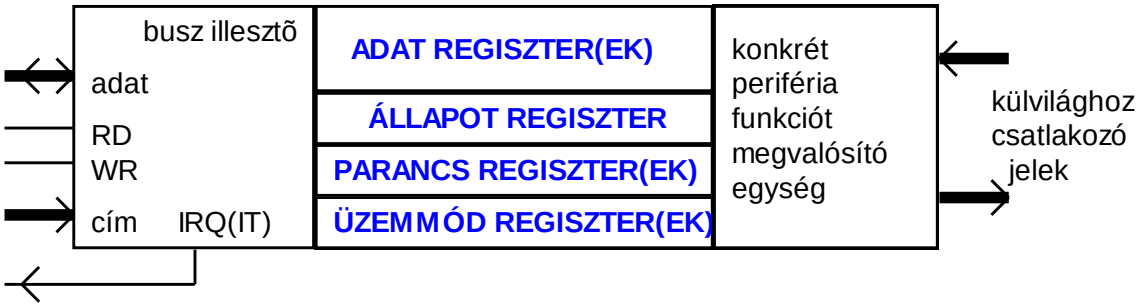
Periféria kezelési módszerek:
Programozott, interruptos, (DMA-s)

Benesóczky Zoltán



A periféria kezelés során **információt kell átvinni a periféria és a memória között**.

A perifériák tipikus regiszterei:



állapot (státus): pl. kész vagy foglalt
parancs: pl. A/D konverter indítás
üzemmód: pl. Timer ciklikusan vagy 1-szer jelezzen
adat pl. soros vonalon (USRT-n) kiírandó vagy beérkezett karakter

Periféria kezelési módszerek

Passzív perifériák

Mindig kész a működésre **bármikor írható ill. olvasható**. Pl. LED display regisztere.

Aktív perifériák

Nincs mindig kész állapotban ezért, ha elkészül jelzést ad. (státus regiszterben, interrupt-tal)

Aktív perifériák kezelése

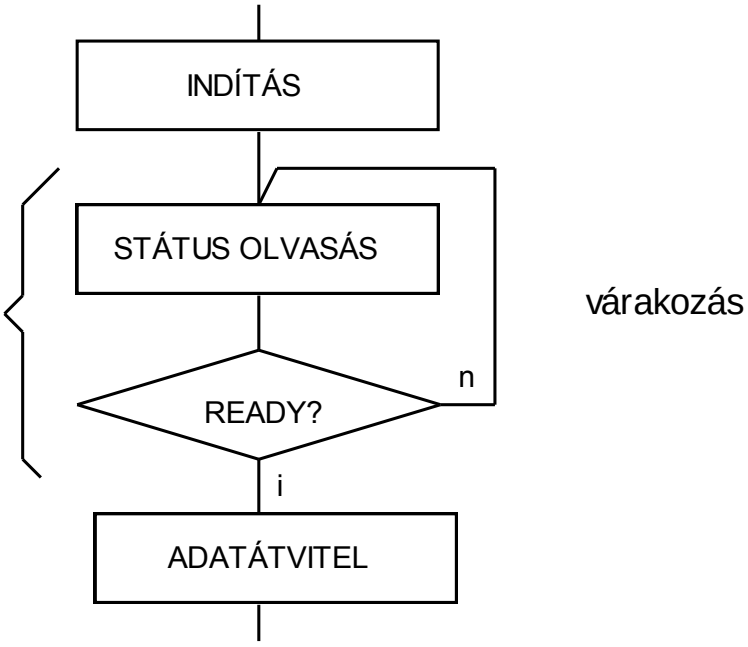
- Programozott státus lekérdezéssel
- Megszakítással (interrupt)
- (DMA-val, nem része az anyagnak)

Programozott státus lekérdezés (polling)

A periféria indítást igényel és az indítás után "rövid idő múlva" készül el, (pl. gyors A/D konverter)

A processzornak a periféria indítása után meg kell várnia, hogy az elkészüljön. **Ha a periféria a processzorhoz képest elég gyors (néhányszor 10 utasítás alatt elkészül)**, akkor megengedhető, hogy a program egy ciklusban addig várjon, míg a státus regiszterből kész jelzést kap, s csak utána folytatja a periféria műveletet (pl. olvassa ki az adatot).

Programozott státus lekérdezés (adatátvitel, ha a státus jelez)

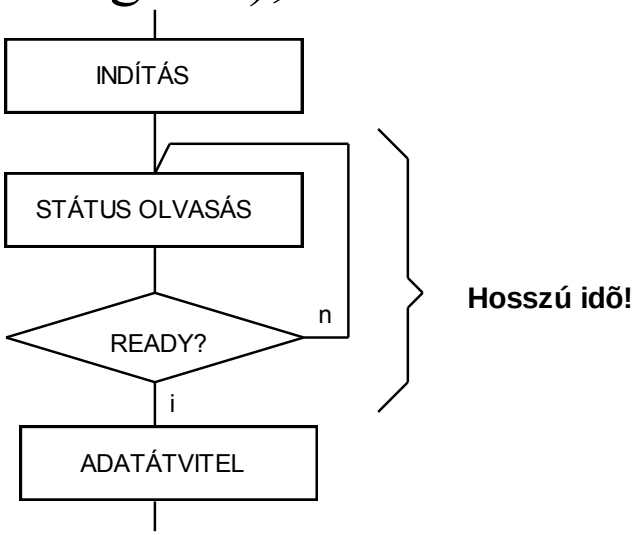


Hogyan lehet hatékonyan kezelni az alábbi tulajdoságú perifériákat?

a. A periféria véletlenszerű időpontban termel adatot (pl: USRT vétel)

b. A periféria indítást igényel és az indítás után “hosszú idő múlva” készül el (pl. lassú A/D konverter)

A státus információra (periféria kész) mindenképpen szükség van. Azonban a lekérdezéses várakozási ciklus nagyon hosszú lehet (ezalatt a uP nem tud más hasznosat végezni), ezért nem hatékony.



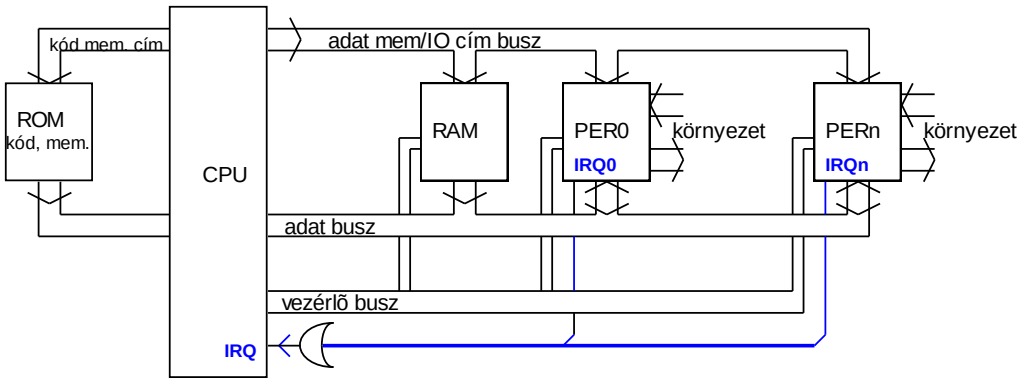
Rossz a CPU kihasználtsága!
Több periféria esetén adat veszhet el!

Interruptos periféria kezelés

- Ha a perifériának kiszolgálásra van szüksége, ezt maga jelzi a CPU-nak.

- A CPU az aktuális utasítás végrehajtása után **elmenti a következő utasítás címét** és **a perifériát kiszolgáló program** (ISR Interrupt Service Routine) **kezdetére ugrik.**

- **A periféria kiszolgálása után a megszakított programot folytatja**.



Külön vonal kell a jelzéshez:
IRQ (interrupt kérés), ezen jelzik a perifériák a CPU-nak a megszakítási igényt. (A fenti ábra egyszerű IT rendszert mutat.)

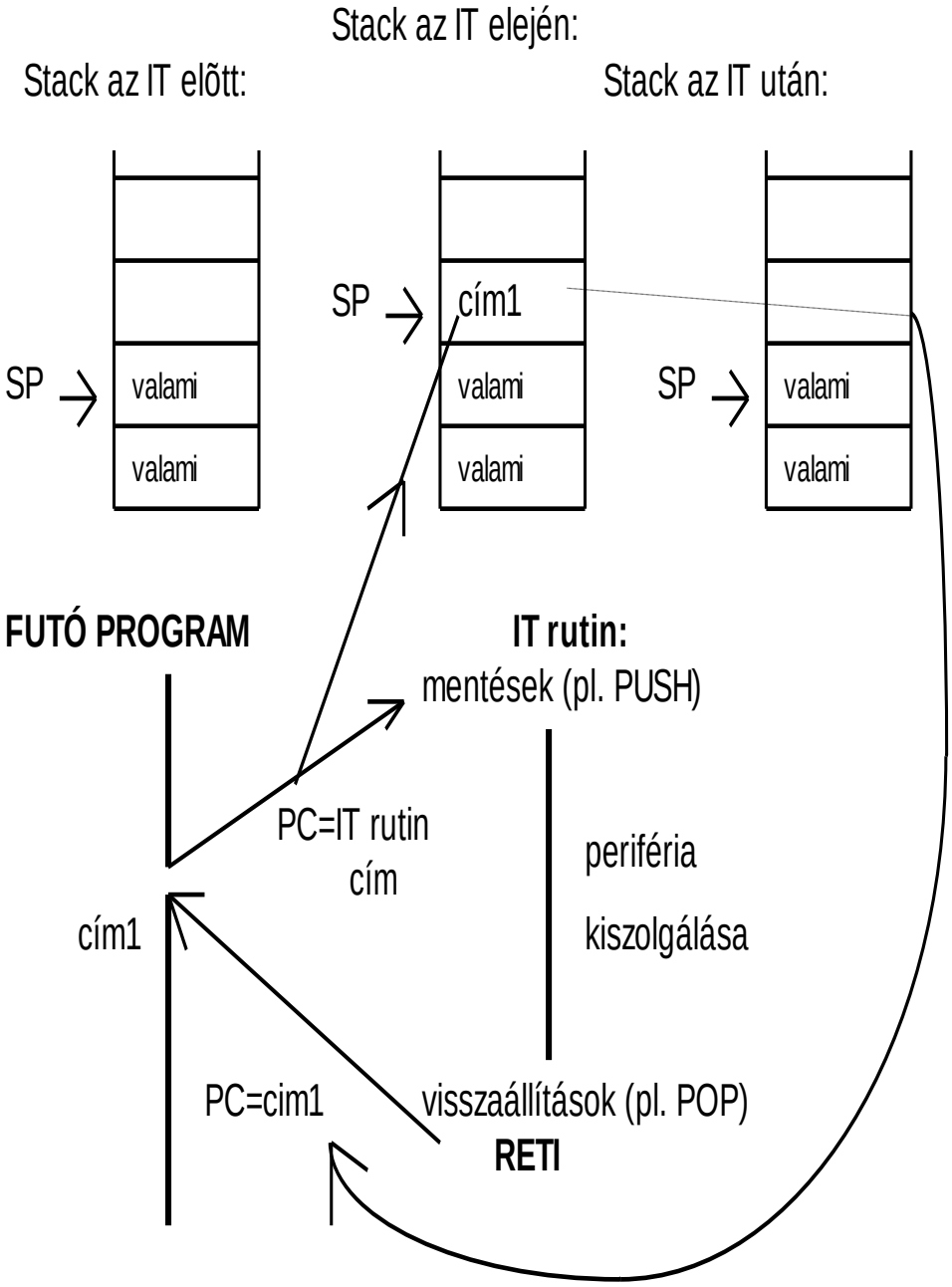
A CPU a kérés hatására abbahagyja az aktuális program futását:

- **Az aktuális utasítás befejezése után**, a perifériát kiszolgáló **ISR–re ugrik**.

- Meg kell jegyeznie, hogy az interrupt program végrehajtása után, hol kell folytatnia a működését. Ezért **az IT rutinra ugrás előtt a stack-re menti a következő utasítás címét** (STACK <= PC) és **van olyan CPU amely a flageket is**. (Pl: MiniRISC)

- A CPU-k egy része (az egyszintű IT rendszerűek) **IT elfogadásakor letiltja a globális IT elfogadást** (IE bit), más CPU-k (a többszintű IT rendszerűek) nem.

- Az IT rutin befejezése után visszatér a megszakított programba. Az **IT rutin végén a RETI** (RTI) **utasítás hatására leemeli a stack tetejéről a visszatérési címet és oda adja a vezérlést** PC <= (STACK).



Az IT rutin nem ronthatja el a megszakított program regisztereit!

- **Az IT rutin elején el kell menteni a használni kívánt regiszterek tartalmát, a végén pedig vissza kell állítani.**

- Az IT rutin feladata a periféria jelzés hatására az **IT rutinban gyorsan elvégezni a feladatok időkritikus részét** (pl. adat elvétele). A többi kapcsolódó (nem időkritikus) feladatot a főprogramra célszerű hagyni.

- A **visszatérés előtt vissza kell állítani a használt regiszterek régi tartalmát**.

- **A mentés a stackre,** vagy valamely **regiszter bankba** történhet, esetleg az IT rutinhoz rendelt **memória területre** (pl. Mini RISC). Magasszintű nyelvek (pl. C) fordítói automatikusan elintézik.

- **A regiszter mentés/visszaállítás vagy egy része lehet automatikus** (a CPU program nélkül elvégzi). Pl. MiniRISC esetén IT hatására a PC mellett a FLAG regiszter automatikusan a stack tetejére mentődik, RTI-re pedig onnan vissza.

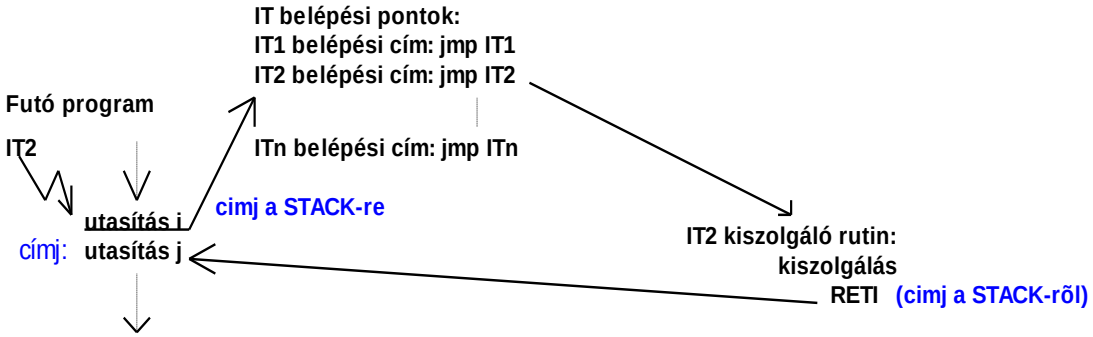
Az IT rutin kezdőcímének és IT forrásának a meghatározása

Honnan tudja a processzor, hogy mi a címe az IT kiszolgáló rutinnak?

- **Egyszerű IT rendszer**
  - **Egyetlen közös IT rutin kezdőcím minden IT forráshoz**.
    Az IT vonalhoz előre definiált nem megváltoztatható kezdőcím tartozik. (Pl. **MiniRISC: 0x01**). Ezen a belépési ponton kezdődhet az IT rutin, vagy az IT rutinra ugró utasítást kell ide tenni**.**
  - **IT forrás (melyik periféria kérte) meghatározása az IT rutinban történik, programozott státus lekérdezéssel.**

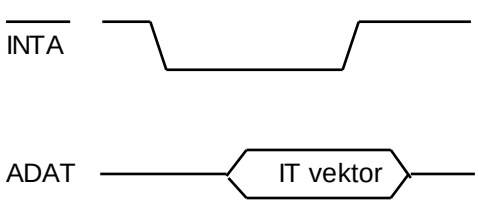
Vektoros IT rendszer

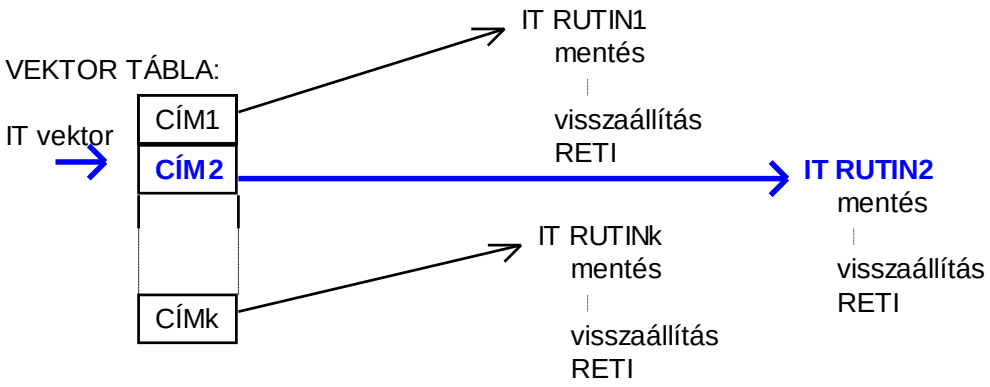
- **Minden IT forráshoz külön IT rutin belépési cím tartozik, tehát IT forrás meghatározása automatikus.**
  - **IT forrásonként fix kezdőcím (IT vektor)**
    (Az egyszerűbb mikrokontrolle-reknél többnyire ezt használják.) **A kezdőcímek általában a RESET címe (többnyire 0) után helyezkednek el a kódmemóriában. A belépési pontokra (címekre) a megfelelő IT rutinra ugró utasítást kell tenni.**



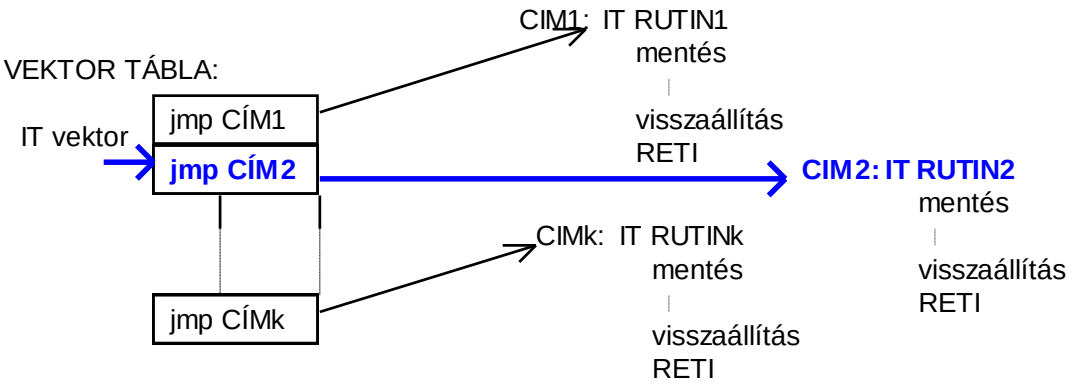
- **Vektor beolvasás**
  A CPU az IT elfogadásakor az INTA (IT elfogadási) ciklusban egy IT vektort vár az adatbuszon.
  A vektort az IT forrás (perifériában levő elosztott IT vezérlő) vagy a központi interrupt kontroller adja.
- **Az IT vektor az egy olyan táblázatba mutató pointer, amelyben az egyes IT rutinok kezdőcímei vannak eltéve.**

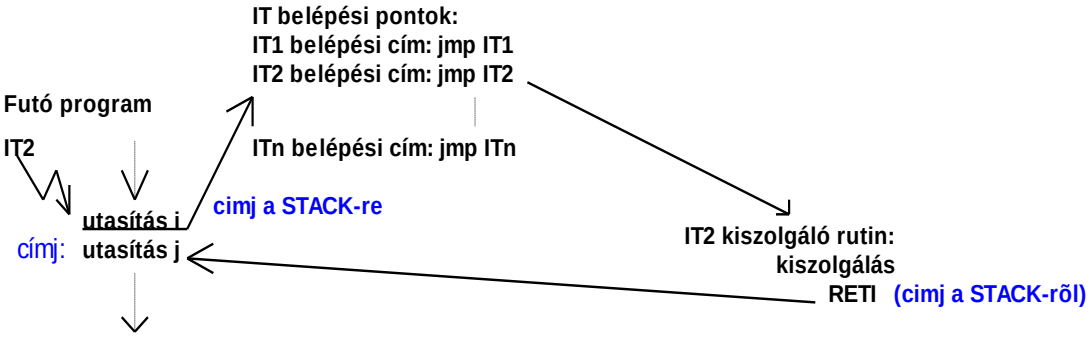
- A processzor az IT vektor alapján az **IT vektor táblából automatikusan előveszi a megfelelő IT rutin kezdőcímet, majd oda ugrik.** (Nem kell hozzá külön ugró utasítás.)





- Sok uC **a kód memóriában elhelyezkedő IT vektor táblának az IT vektor által kijelölt címére ugrik, ahonnan utasítást olvas és azt végrehajtja. Ide az IT rutinra ugró utsítást kell elhelyezni.**



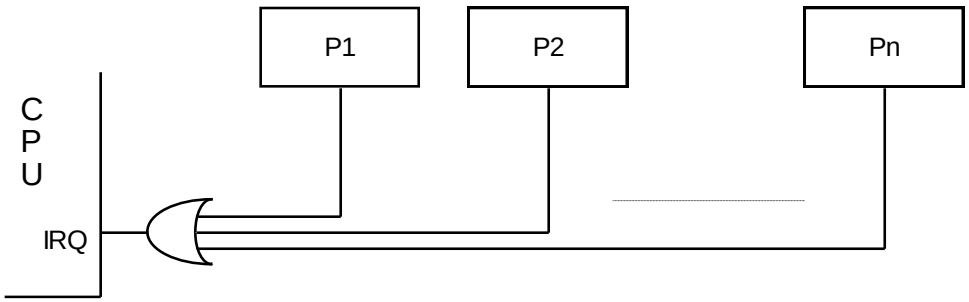


Az interrupt forrás azonosítása

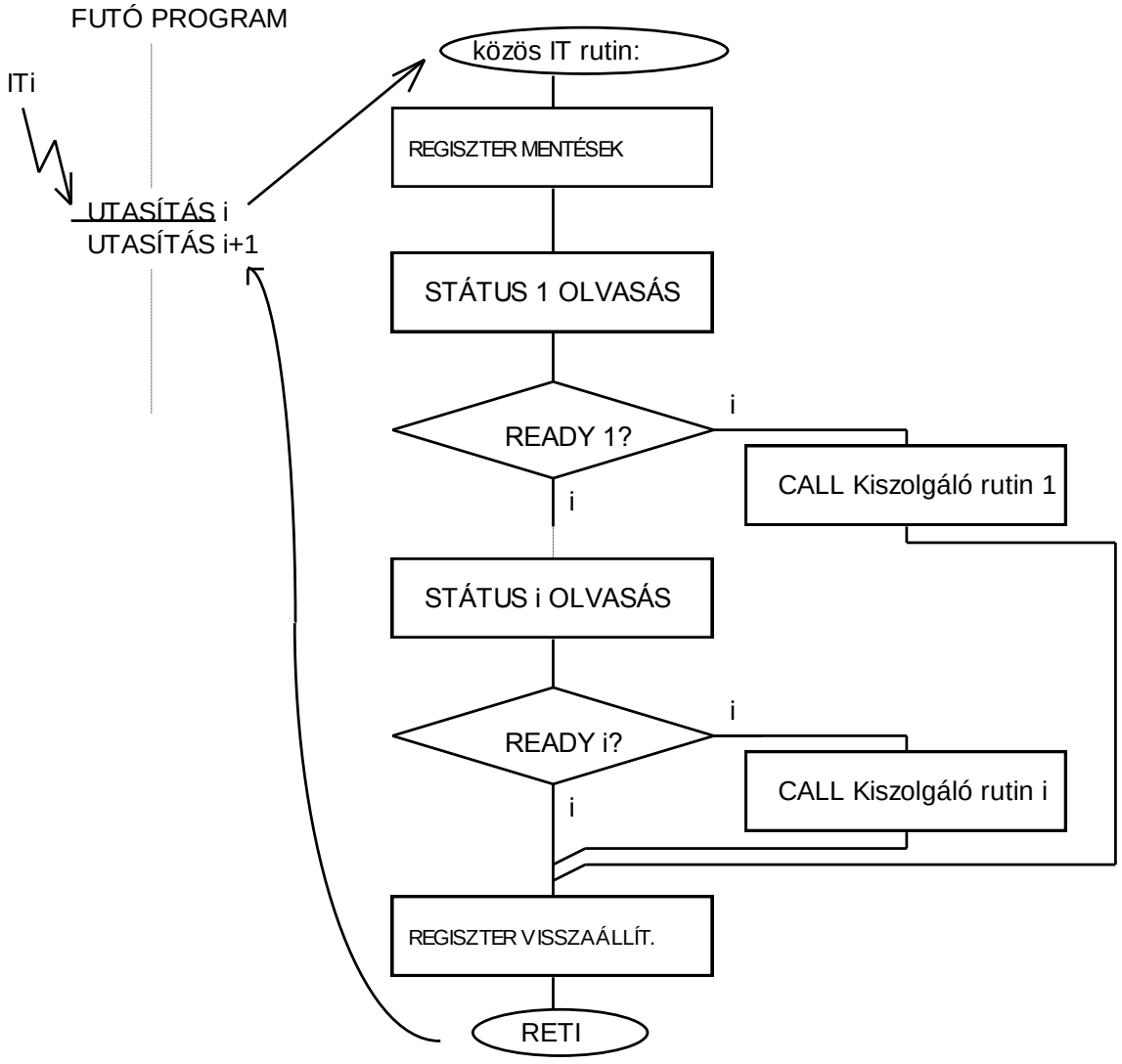
Az interrupt kiszolgáló program kezdőcímének azonosítása nem minden esetben jelenti, hogy egyben az interrupt forrása is ismert.

Programozott (státus) lekérdezéses IT forrás azonosítás

Egyszerű IT rendszer esetén a processzor csak annyi információt kap, hogy valamely periféria IT-kér (akár több is, de ezt nem tudja).



Mivel minden IT esetén ugyanarra a címe adódik a vezérlés, az összes IT forrásnak egyetlen közös IT rutinja van. Ezért ezen belül **programozott státus lekérdezéssel kell azonosítani az IT kérő perifériát. Tehát minden IT-sen kezelt periféria státusregiszterét ellenőrizni kell. A prioritást a lekérdezés sorrendjével lehet megadni.**



A Mini RISC IT rendszere

- Egyszerű IT rendszer
- Minden IT rutin a 0x01 címen kezdődik

  MiniRISC utasítás feldolgozás 19. old

  Időd iagam MiniRISC 72. oldal

Mintaprogram TIMER IT-s kezelésére

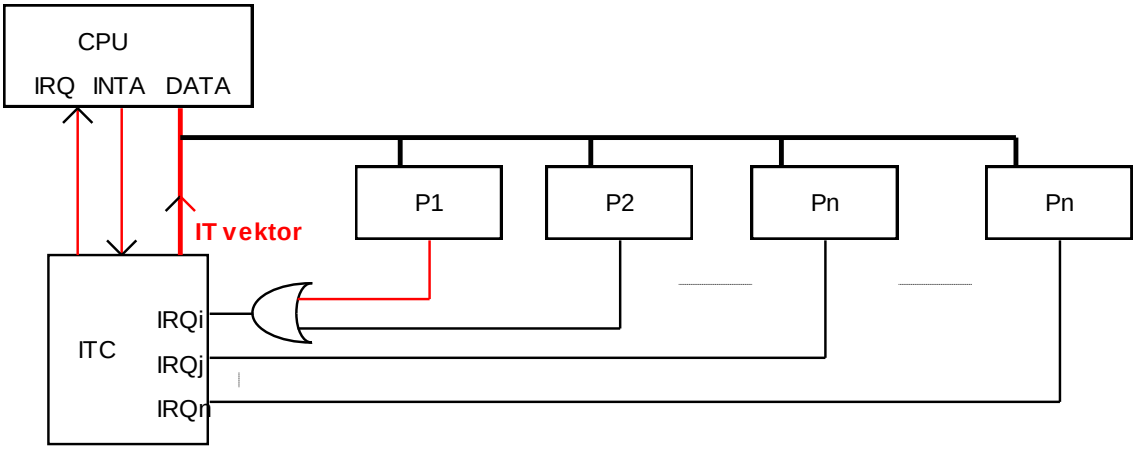
```
DEF LD  0x80            ; LED adatregiszter          (írható/olvasható)
DEF TR  0x82            ; Timer kezdőállapot regiszter (csak írható)
DEF TM  0x82            ; Timer számláló regiszter    (csak olvasható)
DEF TC  0x83            ; Timer parancs regiszter     (csak írható)
DEF TS  0x83            ; Timer státusz regiszter     (csak olvasható)
;7 6 5 4 3 2 1 0
;TIE TPS[2:0] - - TREP TEN
; 1   111     0 0   1   1
DEF TCINI 0b11110011
;TIT TPS[2:0] 0 TOUT TREP TEN
DEF TOUT_MASK 0b00000100

   code
RESET:
   jmp main             ;átugorjuk az IT rutint

ISR:mov r10, TS
   tst r10, #TOUT_MASK
   jz ISR_END
   mov r10, LD
   xor r10, #0x01
   mov LD, r10
ISR_END:
   rti

main:                   ;főprogram belépési pont
   mov r0, #121
   mov TR, r0           ;kezdőérték regiszter
   mov r0, #TCINI
   mov TC, r0           ;parancs regiszter
   mov r0, #0
   mov LD, r0           ;LED-ek kikpacsolása
   sti                  ;globális IT engedélyezés
loop:                   ;üres végtelen ciklus
   jmp loop
```

Automatikus megszakítási forrás azonosítás

Ilyen a **vektoros IT rendszer** ha minden IT forráshoz külön IT rutin cím tartozik. Ilyenkor a megfelelő IT rutinra adódik a vezérlés.

Azonban, **ha több IT forráshoz közös IT vektor tartozik akkor itt is programozott státus lekérdezéssel kell azonosítani a kérés forrását**.

Ezt vegyes IT rendszernek nevezik.



A **vegyes IT rendszer** esetén tehát egyes IT források automatikusan, mások lekérdezéssel azonosíthatók.

Az interrupt engedélyezése

- **Globálisan CPU utasítással** tiltható, engedélyezhető (CLI, STI vagy DI, EI)
- **A periféria parancs regiszterében**
- Az **interrupt vezérlőben** (ahol van ilyen)

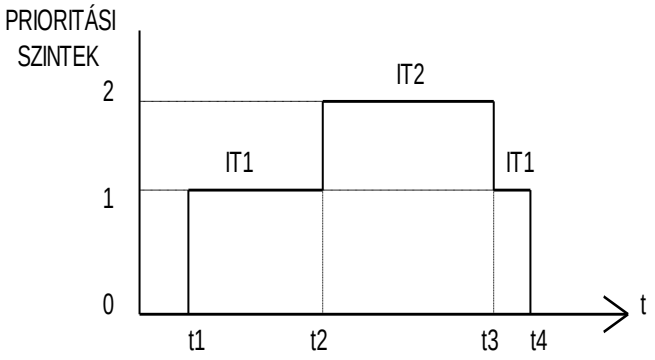
NMI (Non Maskable Interrupt)

- Nem, vagy csak speciálisan tiltható.
- A **globális IT tiltó és engedélyező utasítás nem hatásos**.
- **Más interrupt nem képes megszakítani.**
- A legfontosabb (kritikus) események jelzésére használják
- A külső NMI bemenet (ha van) **élérzékeny**

Az interrupt rutinok megszakíthatósága

- ***Egyszintű interrupt rendszerben***
  Az IT rutinok nem szakíthatják meg egymást. Az egyszerű IT rendszer egyben egyszintű is.

- ***Többszintű vektoros interrupt rendszerben*** a **magasabb szintű megszakíthatja az alacsonyabbat. Azonos szintű kérések esetén a szinten belüli magasabb prioritású szakíthatja meg az alacsonyabbat.**
  Többnyire 2 vagy 3 szint van. (Pl. Atxmega uC: Low, Medium, High)
  Az IT források szintjét felprogramozáskor szokás megadni, de a program megváltoztathatja. Pl: a főprogramot



Prioritási stratégiák

Az egyszerre érkező kéréseket milyen sorrendben szolgálja ki a CPU?

A legnagyobb prioritásút először.

- **Fix prioritás** esetén a prioritás nem változtatható.

- **Változó prioritás** esetén rugalmasan változtatható
  Pl. ***Körben forgó prioritás*** esetén a prioritási sorban legelöl állóból (legnagyobb prioritásúból) a kiszolgálása után a legkisebb prioritású válik

Az IT kérő jel

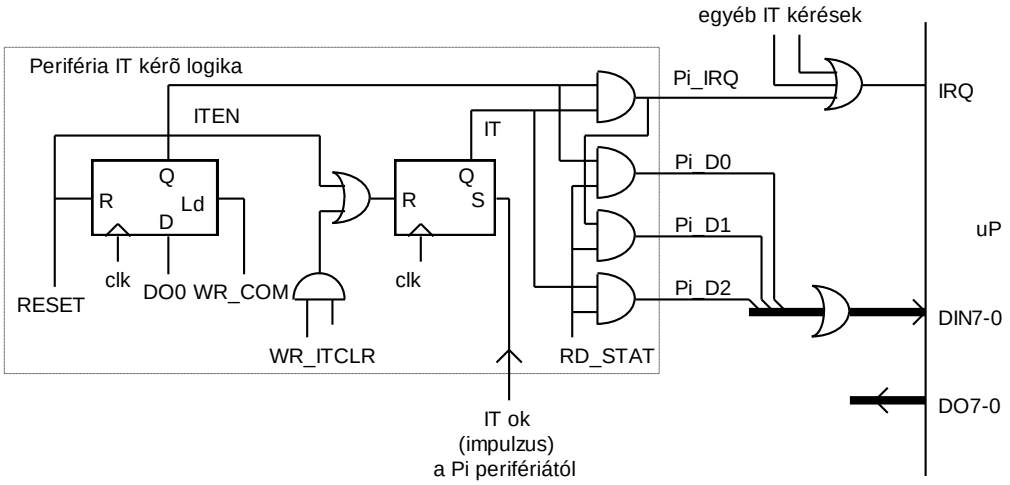
- IRQ vonal lehet **szintérzékeny** és **élérzékeny**

- **Szintérzékeny esetben a kiszolgálásig fenn kell tartani a kérést.**

Törlése lehet:

- **Az IT rutinban programból. (A státus olvasása törli, vagy külön írással kell törölni a bitet.)**
- **A periféria törli automatikusan, amikor az IT kérését elfogadja a CPU.**

IT-s periféria kialakítása a MiniRISC processzornál



Interrupt vezérlő kialakítása

Az interrupt vezérlő lehet önálló egység (centralizált), vagy elhelyezkedhet a perifériákban elosztva (decentralizált).