BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM
VILLAMOSMÉRNÖKI ÉS INFORMATIKAI KAR
MÉRÉSTECHNIKA ÉS INFORMÁCIÓS RENDSZEREK TANSZÉK

# Digitális technika VIMIAA02 12. hét

Fehér Béla, Benesóczky Zoltán
BME MIT

# Interruptos periféria kezelés II.

**Megszakításrendszerek feladatai:**

- **Megszakítási események jelzése**
  *Minden periféria* ***külön jelvezetéken*** *ad jelzést.*
  - ***Szint érzékeny*** *Az IT kérés a jel szintjének hatására történik:* a kérés történhet a jel *L szintjére* vagy *H szintjére* (hardver függő)
  - ***Élérzékeny*** *Az IT kérés a jel élének hatására történik:* a kérés történhet a jel *felfutó élére* vagy *lefutó élére*, (hardver függő).

  *Az IT kérés legtöbb esetben szintérzékeny. Ekkor a kiszolgálásig fenn kell tartani a jelzést.* ***A kérés törlését az IT kiszolgáló program végéig el kell végezni.*** (Pl. MiniRISC esetén H szintet kell adni a kiszolgálás elkezdéséig.)

  A különleges események jelzésére fenntartott nem maszkolható (nem tiltható) interrupt (*NMI*) mindig élérzékeny. (NMI-vel nem rendelkezik minden CPU.)

# Interruptos periféria kezelés II.

**Megszakításrendszerek feladatai** (folytatás):

- **Megszakításkérés prioritásának kezelése**
  Melyik perifériát ***szolgálja ki először, ha egyszerre több periféria kér IT-t?*** A kérők közül ***az aktuálisan legnagyobb prioritásút***.
  - Az *NMI* prioritása a legnagyobb.
  - A többi esetében többféle ***prioritási stratégia*** alkalmazható.
  - **Fix prioritás**
    A prioritás előre meghatározott, nem változtatható.
  - **Változó prioritás**
    A prioritás rugalmasan változtatható.
    Pl. ***Körben forgó prioritás*** esetén a prioritási sorban legelöl állóból (legnagyobb prioritásúból) a kiszolgálása után a legkisebb prioritású válik

# Interruptos periféria kezelés II.

**Megszakításrendszerek feladatai** (folytatás):

- **Megszakításkérések engedélyezése / tiltása**
  - **A processzorban** *a **globális IT elfogadás engedélyező bit** 1-be írásával kell engedélyezni, 0-ba írásával tiltani.* ***A RESET letiltja.***
    *Ezt a processzor megfelelő utasításai végzik.*
    ***Engedélyező** utasítás elnevezései:* ***sti, ei***.
    *A **tiltó** utasítás elnevezései:* ***cli, di***
    **A *különleges események*** *(pl. tápellátás probléma)* jelzésére fenntartott ***NMI*** (nem maszkolható interrupt) ***nem tiltható le***.
  - **A perifériában** az IT engedélyező bit(ek)el engedélyezhető. Egy periféria több IT eseményt is generálhat, amelyek külön engedélyezhetők. ***A RESET letiltja.***
- **Megszakításforrás azonosítása**
  *Honnan tudja a processzor, hogy **melyik periféria kérte az IT-t**? IT rendszer függő, ott tárgyaljuk.*
- **Az ISR kezdőcímének meghatározása.**
  ***Milyen címre kell ugrani az IT elfogadásakor?*** *IT rendszer függő, lásd ott.*

# Interruptos periféria kezelés II.

- **Megszakításrendszerek feladatai** (folytatás):
  - **Megszakításkérés kiszolgáló szubrutin (ISR) végrehajtása, a szükséges adminisztráció elvégzése.** *Milyen kötelező elvégzendő feladatok vannak az IT rutinban?* ***El kell menteni*** *(meg kell őrizni)* ***az IT rutinban használt regiszterek értékét*** *(és a FLAG-ek értékét is).* A mentés a **stack-be**, vagy valamely **regiszter bankba** (alternatív regiszter készlet) történhet. (A MiniRISC-knek nincs stack-be adatot író és onnan adatot olvasó utasítása és csak egy regiszter készlettel rendelkezik.)
    Egyes uC-k több regiszter készlettel (bank) rendelkeznek, melyek között gyorsan lehet váltani.
  - *El kell végezni a periféria kezelését és az egyéb elvégzendő feladatokat. (Periféria és feladat függő.)*

# Interruptos periféria kezelés II.

- **Megszakításrendszerek feladatai** (folytatás):
  - **Visszatérés az IT rutinból a program eredeti állapotának visszaállításával.**
  - *Vissza kell állítani a CPU IT előtti állapotát. Pl.* ***A megszakított program által használt regiszterek és a flag regiszter értéke ugyanaz kell, hogy legyen, mint a megszakítás előtt.***
  - *Ha a mentés regiszter bank váltással történt, akkor* ***vissza kell váltani az eredeti regiszter bankra****.*
  - *Ha a mentés stack-re írássokkal történt (push ri, push rj,...), akkor fordított sorrendben* ***vissza kell olvasni a stack-ből a regiszterekbe az adatokat*** *(..., pop rj, pop ri.).*

# Interruptos periféria kezelés II.

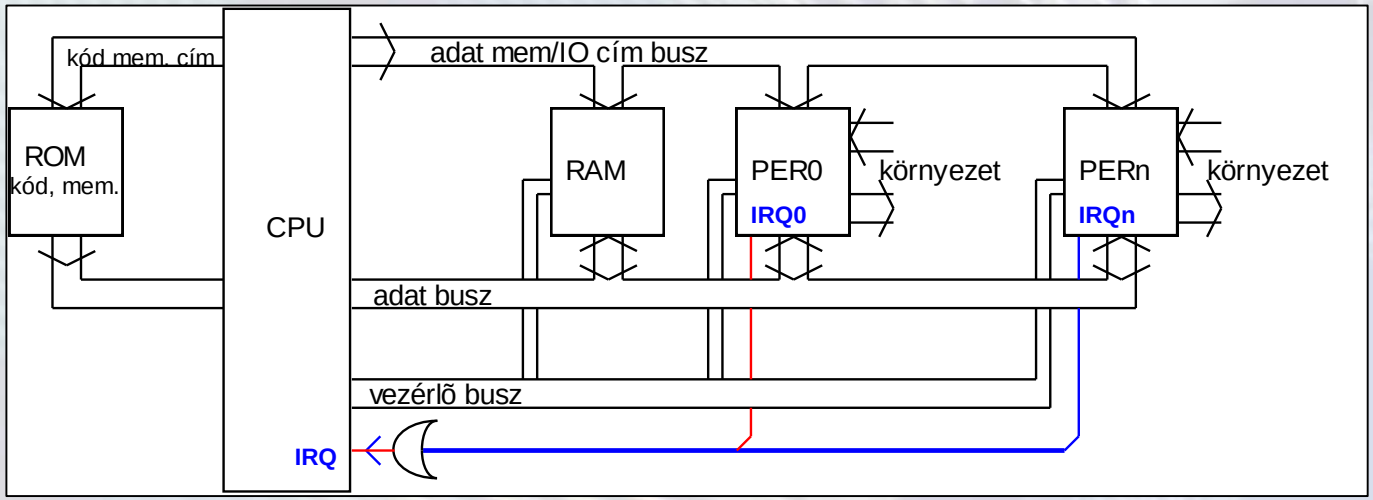
- **Megszakításrendszerek feladatai** (folytatás):
- ***A regiszter mentés/visszaállítás, vagy egy része lehet automatikus*** (a CPU program nélkül elvégzi). Pl**. *MiniRISC*** esetén IT hatására ***a PC mellett a FLAG regiszter automatikusan a stack tetejére mentődik*, *rti***-re pedig onnan ***vissza***.
- A ***MiniRISC processzor esetén*** a mentés/visszamentés helyett ***a regiszter készletet 3 részre osztjuk***. Az egyik részt ***csak a főprogramban használjuk***. A másikat ***csak az IT rutinban***. A harmadikat pedig ***adatok átadásra a főprogram és az IT között*** (ha van ilyenre szükség). (Nem írunk olyan programot, amely ennél komlexebb megoldást igényel.)
- ***A kiszolgálás végéig meg kell szüntetni a kiszolgált periféria IT kérését. Ennek módja periféria és IT rendszer függő.***

# Interrupt rendszerek

- **IT rendszerek osztályozása.**
  - **Egyszerű IT rendszer**
    ***Minden IT-nek*** **ugyanaz** ***az ISR belépési pontja*** (címe).
  - **Vektoros IT rendszer**
    ***Minden IT forráshoz*** **különböző** ***ISR belépési pont*** (cím) tartozik.
  - **Egyszintű IT rendszer**
    Egy interrupt ***nem szakíthat meg*** egy ***másik interruptot***.
  - **Többszintű IT rendszer**
    Egy interrupt **megaszakíthat egy másikat** (alacsonyabb prioritásút)
- ***Az egyszerű IT rendszer egyben egyszintű is.***
- ***A vektoros IT rendszer lehet egy és többszintű is.***

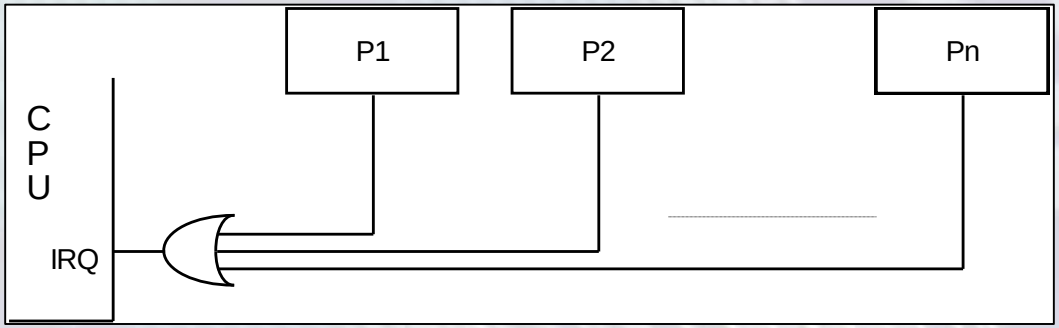
# Egyszerű IT rendszer

- **Egyszerű IT rendszer**
  - Egyetlen ***közös IT rutin kezdőcím tarozik minden IT forráshoz.*** (Pl. MiniRISC: 0x01). Ezen a belépési ponton kezdődhet az IT rutin, vagy ide az IT rutinra ugró utasítást kell tenni.
  - ***A processzor IRQ*** (IT kérés) ***bemenetére a perifériák IRQi interrupt kérő jeleinek VAGY kapcsolata kerül***, így az IRQ csak azt jelzi, hogy valamelyik (akár több is) IT-t kér.



# Egyszerű IT rendszer

- **Az IT forrásának a meghatározása egyszerű IT rendszernél**
- Egyszerű IT rendszer esetén tehát a processzor csak annyi információt kap, hogy valamely periféria IT-kér.

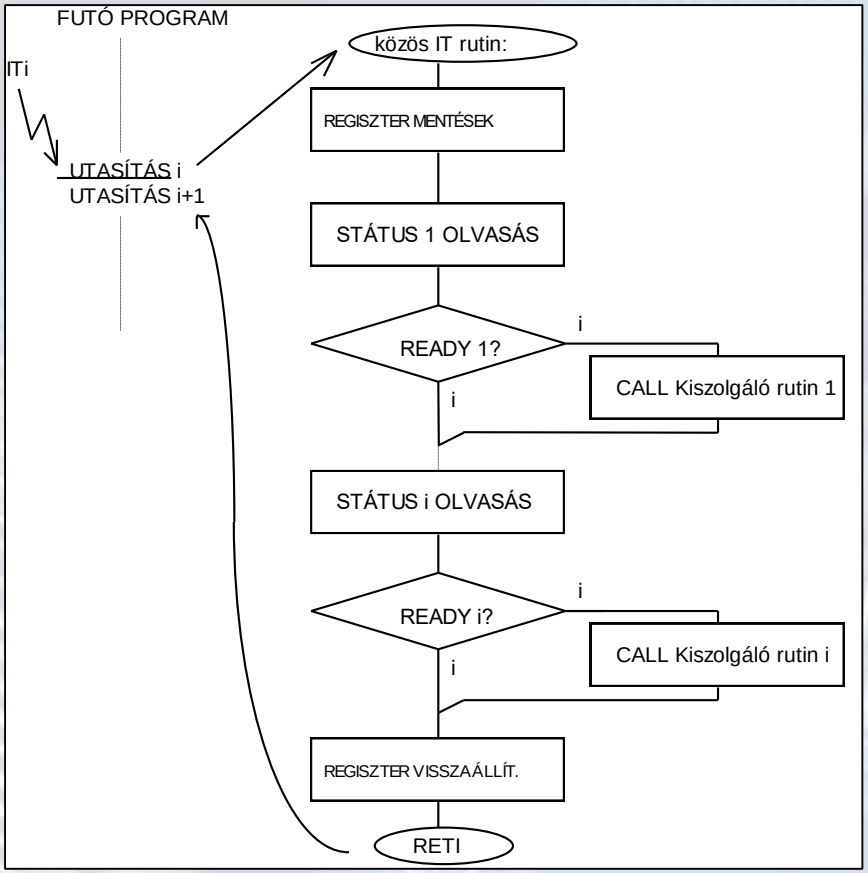



- ***Az IT kérés hatására*** a visszatérési cím stack-re mentése után ***a közös ISR belépési pontra adódik a vezérlés*** és ***a globális IT kérés engedélyezése automatikusan letiltódik*** (az előző, engedélyezett állapot elmentődik.).

# Egyszerű IT rendszer

**Az IT forrásának a meghatározása és kiszolgálása egyszerű IT rendszer esetén.**

- A közös IT rutinban minden IT-sen kezelt periféria státus regiszterének lekérdezésével lehet azonosítani, hogy melyek kértek IT-t.
- A státus lekérdezés sorrendje dönti el a prioritást, vagyis hogy több IT kérő esetén melyik lesz először kiszolgálva.



# Egyszerű IT rendszer

- A ***visszatérés előtt a periféria IT kérését törölni kell.***
- Az IT törlésének módja többféle lehet (periféria függő).
  - A periféria státus regiszer kiolvasása után az ***IT-t jelző bitjébe 1-et kell írni***. (Nem 0-át! Pl. MiniRISC BT periféria BTIF regisztere.)
  - A periféria ***státus regiszterének olvasása törli*** a periféria IT kérését (MiniRISC TIMER periféria).
  - ***Az IT kérő esemény megszűnése törli*** az IT kérést. (Pl. A „FIFO-ban adat van” jelzést a FIFO összes adatának a kiolvasása, a MiniRISC USRT perifériában.)

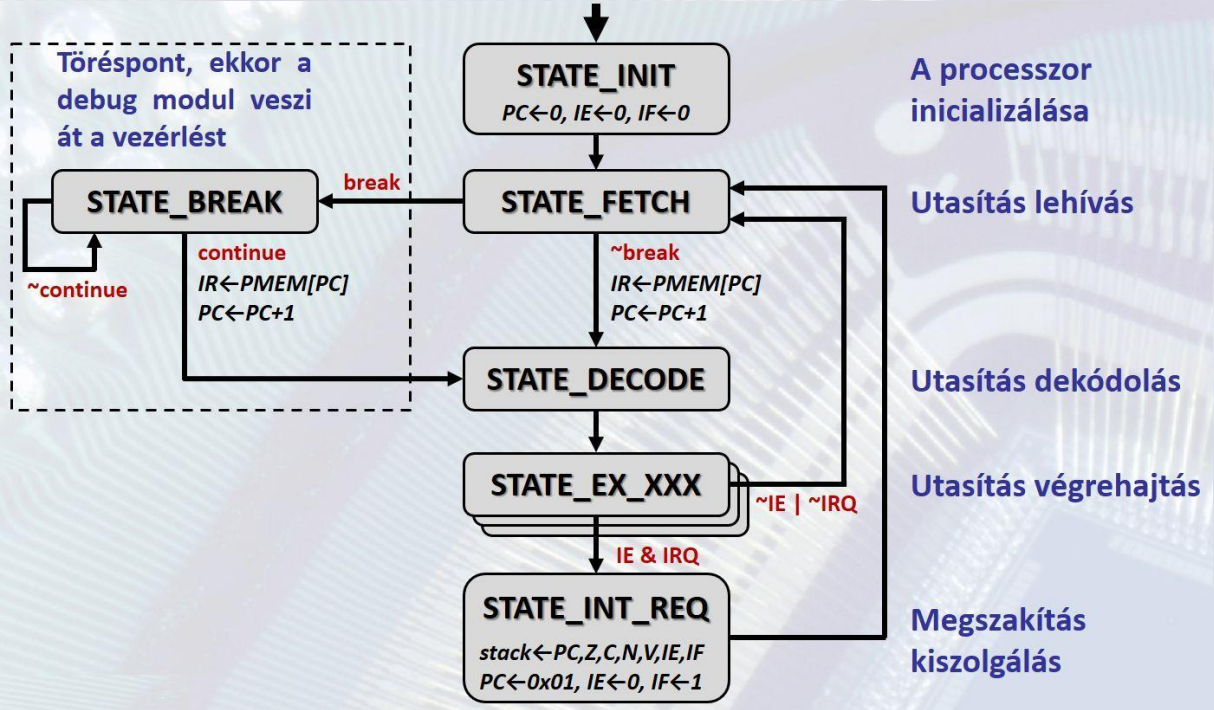
# A MiniRISC megszakításrendszere

- A MiniRISC ***interrupt rendszere egyszerű és egyszintű***.
- A CPU az ISR-re ugrás előtt ***a stack-re menti a megszakított program következő (még nem végrehajtott) utasításának a címét és a flag regiszter tartalmát továbbá letiltja a további interruptokat*** (IE =0 lesz).
- Minden IT hatására a ***0x01 címre adódik*** a vezérlés.
- Több interruptosan kezelt periféria esetén az IT rutinban ***a perifériák státus regiszterének lekérdezésével deríthető ki, hogy melyik periféria kért IT-t***.
- Az interrupt kiszolgáló rutinban ***a visszatérés előtt törölni kell az IT kérő periféria interrupt kérését***. (Az interrupt kérő jel H szint érzékeny.)
- Az IT rutinból az ***rti*** utasítással kell visszatérni. Az ***rti*** utasítás leszedi a stackről a visszatérési címet és a flag regiszter elmentett tartalmát. Utóbbit visszatölti a flag regiszterbe és a visszatérési címre ugrik. A ***visszatérés után az interrupt újból engedélyezve lesz***.

# A MiniRISC megszakításrendszere

**A MiniRISC inetrrupt elfogadása részletesen.**

- A Mini RISC processzor vezérlőjének állapotgépe, ha az utasítás execute állapotában ***az IRQ vonalat 1-ben találja, és az IT elfogadás engedélyezett***, a FETCH állapot előtt ***beiktatódik egy interrupt elfogadás állapot*** (INT_REQ).



# A MiniRISC megszakításrendszere

- **A megszakítás kiszolgálásának folyamata a MiniRISC processzor szempontjából a következő:**
  - A CPU ***észleli az IRQ vonal aktiválását***.
  - A CPU ***az aktuális utasítást végrehajtja.***
  - Ha ***IE = 1 és IRQ = 1***, akkor az ***EXECUTE fázis után belép az IRQ_REQ állapotba***. (Egyébként soha.)
  - ***Az IRQ_REQ állapotban*** lényegében ***végrehajt egy szubrutinhívást a megszakításvektor*** címére (PC=0x01). Egyúttal ***a visszatérési címet és állapotjelző biteket a hardver verembe menti*** (PC,Z,C,N,V,IE,IF)
  - PC = 0x01, **IE = 0** és IF = 1 állapottal végrehajtja az ISR megszakítás kiszolgáló rutinra ugró utasítást. Tehát letiltja a további IT-k elfogadását.

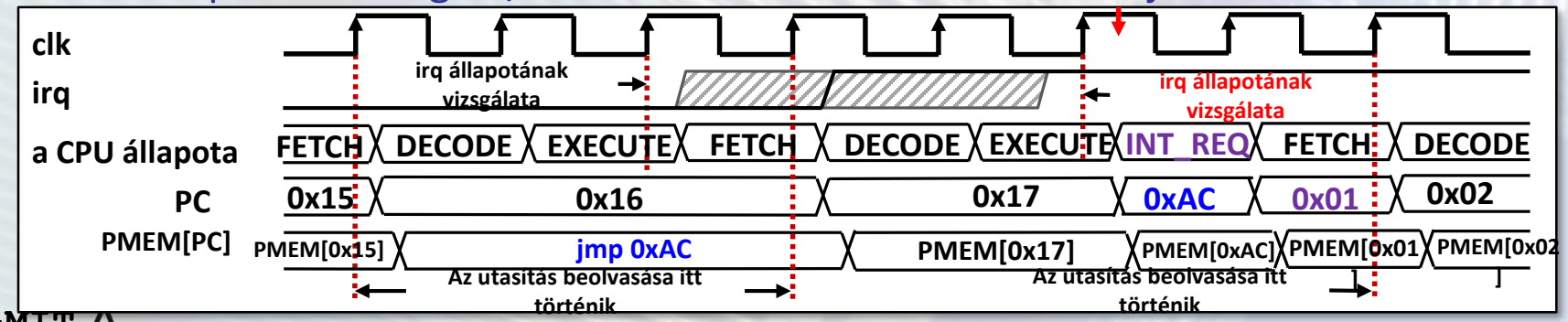
# MiniRISC processzor – IT elfogadás

**Az IT elfogadás időbeli lefolyása**

- ***A PC értéke már a FETCH ciklus végén nő (PC+1 lesz), de ugrás*** vagy ***szubrutinhívás*** esetén a PC értéke módosul(hat) a végrehajtási (execute) fázisban is. (Az alábbi ábra ilyen esetet mutat.)
- Megszakításkérés kiszolgálása esetén ***az INT_REQ állapotban a PC-be betöltődik a megszakítás vektor (0x01), a STACK-be pedig a következő utasítás címe.*** (Az alábbi példában 0xAC.)
- Az aktuális utasítás végrehajtása 4 órajel ciklus hosszú:

  **FETCH, DECODE, EXECUTE, INT_REQ**

  Ugrás/szubrutin esetén is a következő lehívási (fetch) fázisra éppen időben megjelenik az új cím (0x01) a programmemória címbuszán és az INT_REQ állapotban az ugrás/szubrutin címét is a STACK-re tudja menteni.



# MiniRISC processzor – megszakítás

- A mintapéldában a főprogram 0x08-as címén lévő utasítást hajtja végre a CPU, amikor az IT kérést észleli.
- A STACK-re meneti 0x09 en levő utasítás (jsr print_str) címét és az IT belépési pontra ugrik (0x01), majd innen az ISR elejére (0x30: ustr_rx).
- Az ISR-t végrehajtja, majd az ***rti*** utasítás hatására a CPU a stackből kiveszi a visszatérési címet (0x09) és oda ugrik (folytatja a megszakított programot).

```
00:    jmp start
01:    jmp usrt_rx

02:  start:
02:    mov r0, #0
03:    mov LD, r0
04:    mov r0, #0x3b
05:    mov UC, r0
06:    sti
07:  loop:
07:    jsr delay
08:    mov r8, #str
09:    jsr print_str
0A:    jmp loop
```

IRQ

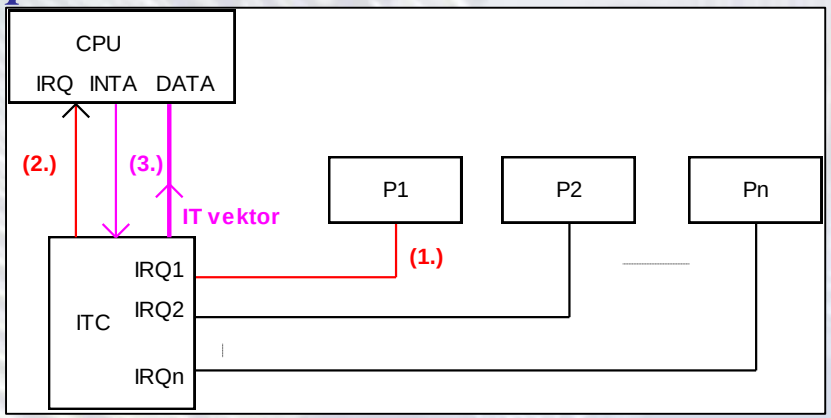
stack←{PC (0x09),flag-ek}
PC←0x01, IE←0

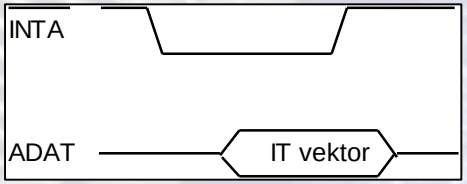
```
30:  usrt_rx:
30:    mov r15, UD
31:    mov LD, r15
32:    rti
```

{PC,flag-ek}←stack

# Vektoros IT rendszer

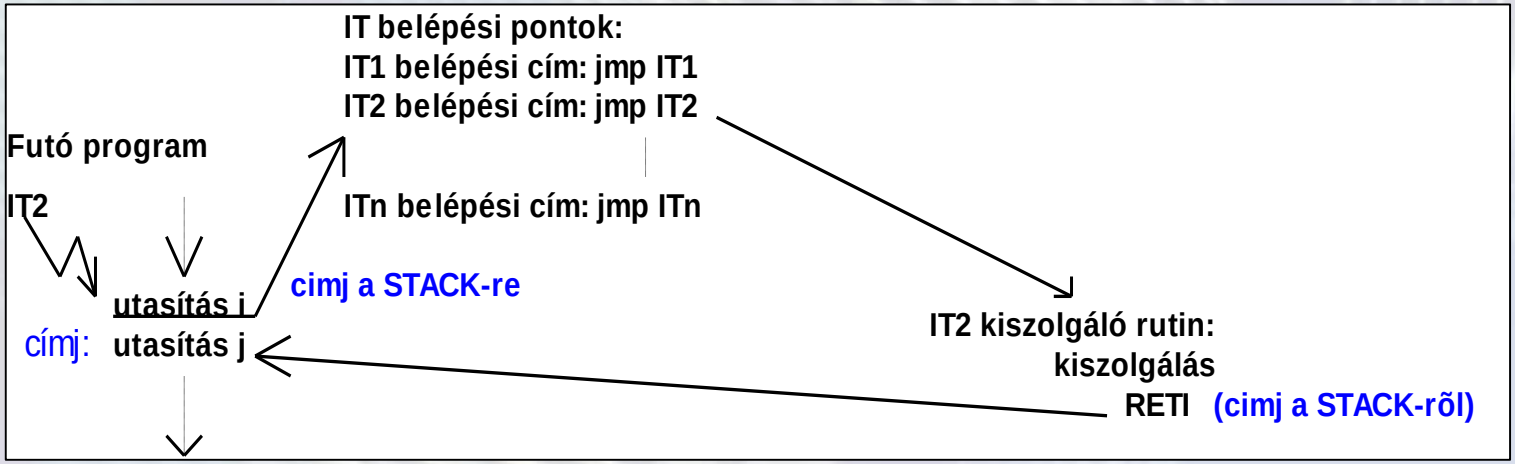
- Az ***egyszerű IT rendszerben az IT forrás meghatározása lassú***, a közös IT rutinban történő státus lekérdezés miatt. Ezt a hátrányt küszöböli ki a vektoros IT rendszer. A vektoros IT rendszerben az IT kérő jelek fogadására egy külön ***IT vezérlőt*** (ITC) használnak.
- ***A vektoros IT rendszernél minden IT forráshoz külön IT rutin belépési cím*** tarozik, így az IT forrás meghatározása automatikus.
- ***A processzor*** az interrupt vezérlőtől az IT elfogadási ciklusban ***az IT forrást azonosító információt (IT vektort) kap.***
  - A periféria aktivizálja az interrupt vezérlő (ITC) hozzá tartozó IRQ vonalát.
  - Az ITC aktiválja a CPU IRQ vonalát.
  - A CPU kiadja az INTA jelet, ami alatt az ITC az adatbuszra teszi az IT vektort, amit a CPU beolvas.



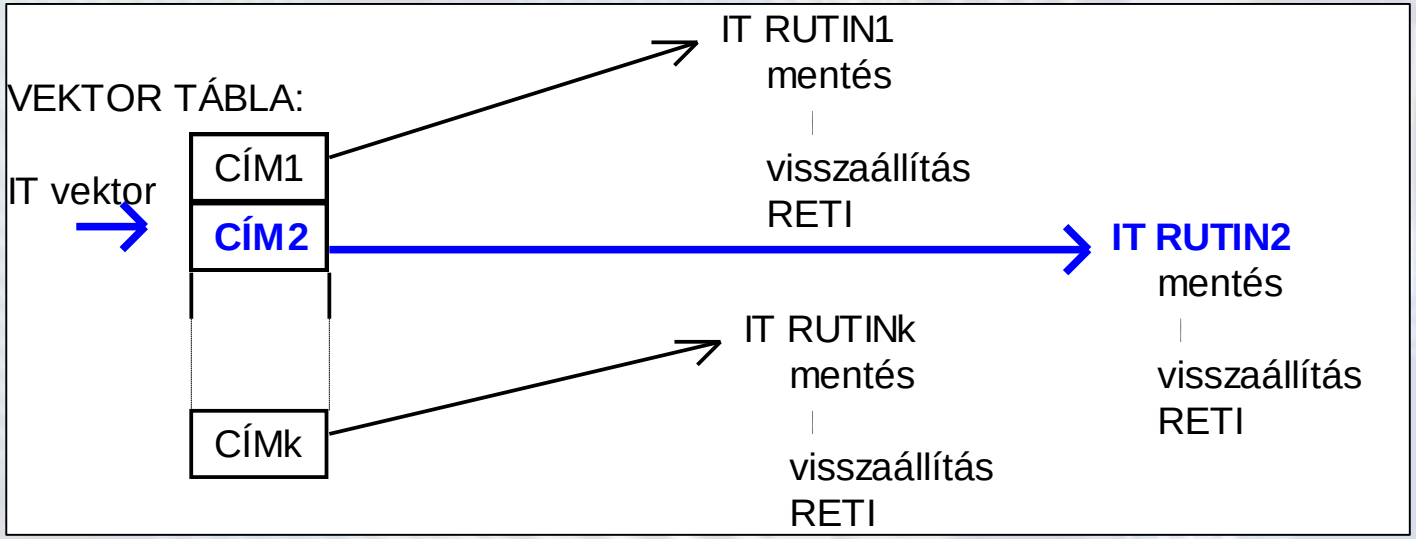
# Vektoros IT rendszer

- ***Az egyik módszer szerint a CPU az IT vektor alapján határozza meg az IT belépési pontot és oda ugrik.***
- ***Az egyes IT forrásokhoz tartozó kezdőcímek általában a RESET belépési pont után helyezkednek el***, közvetlenül egymás után a kódmemóriában. (Két belépési pont között elfér egy ugró utasítás kódja.)
- A belépési pontokra ***a megfelelő IT rutinra ugró utasítást kell elhelyezni.***



# Vektoros IT rendszer

- Egy másik módszernél a CPU által kapott vektor ***egy IT vektor tábla*** valamelyik elemére mutat. ***A táblában az IT rutinok címeit kell elhelyezni.***
- A CPU a megfelelő cím kiolvasása után ***arra a címre ugrik, amit kiolvasott***.

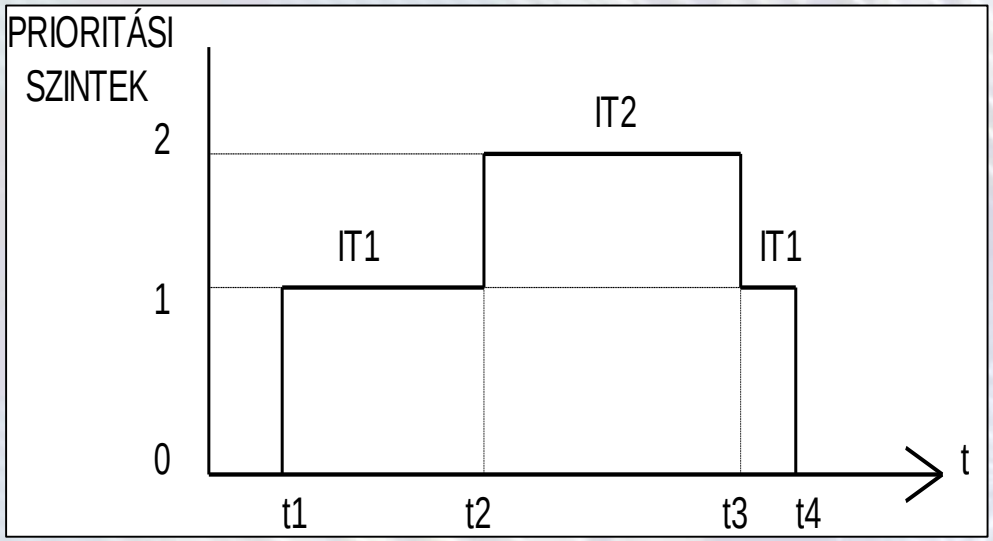


# Vektoros IT rendszer

- A vektoros IT rendszer lehet többszintű is, ekkor egy magasabb prioritású IT megszakíthat egy alacsonyabb prioritásút.
- Azonos szintű kérések esetén a szinten belüli magasabb prioritású szakíthatja meg az alacsonyabbat. Többnyire 2 vagy 3 szint van.
- Az IT források szintjét felprogramozáskor szokás megadni, de a program megváltoztathatja.

# Vektoros IT rendszer

- Az alábbi példában a főprogramot (legalacsonyabb szint) t1 időpontban megszakítja IT1.
- Az IT1-et a t2 időpontban megszakítja a nála magasabb prioritású IT2.
- IT2 t3 időpontban visszatér és folytatódik IT1.
- IT1 t4 időpontban visszatér és folytatódik a főprogram.

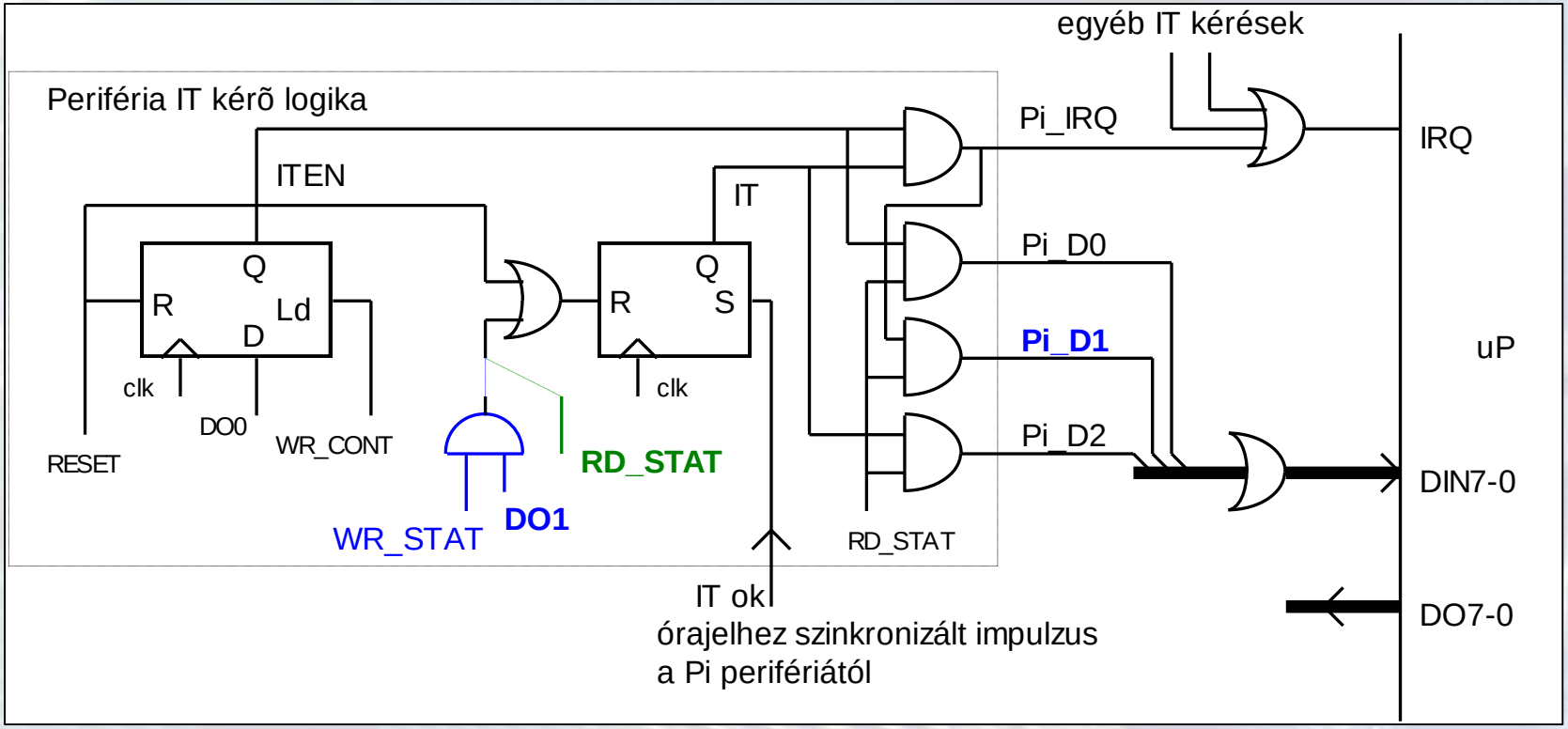


# Programszervezési alternatívák

- **A megszakításrendszer tervezése a programszervezésre is hatással lehet**
  - A programok feladatai megoszlanak a főprogram és a periféria kiszolgálást végző ISR között.
  - Ennek szintjei a következők:
  - Az ISR csak a legszükségesebb perifériaműveleteket hajtja végre. Minden más a főprogramban történik.
  - Az ISR a minimális aktivitáson túl általános feladatokat is elvégez, a perifériához kapcsolódóan.
  - Az ISR tartalmazza az összes szükséges feladatot, a főprogram szinte egy aktivitás nélküli várakozó hurok. (Ez utóbbi nagyon ritkán használt szélsőség.)

# Interruptos periféria illesztése

## Interruptos periféria illesztése MiniRISC processzorhoz




A cím és parancs dekódert (RD_STAT és WR_STAT) előállítását nem rajzoltuk fel.

# Interruptos periféria illesztése

**Interruptos periféria illesztése MiniRISC processzorhoz**

- A perifériától jövő eseményt egy 1 órajel hosszú impulzus jelzi (pl. időzítés letelt).
- Az impulzus 1-be állítja az eseményt jelző státus bitet (IT).
- Az interrupt kérést egy programból írható üzemmód bit (interrupt engedélyező bit, ITEN) engedélyezi.
- Interrupt kérés akkor lesz, ha a Pi_IRQ =ITEN*IT = 1.
- Az IT kérést törlését okozhatja a státus regiszter visszaolvasása (*RD_STAT*), vagy okozhatja a státus regiszter címére írt, a törlendő IT bit helyén 1-et tartalmazó adat (***WR_STAT*DOi***). Mindkét megoldást felrajzoltuk.
- Az esemény jelző IT bitet és az ITEN bitet a RESET törli.

# MiniRISC IT minta program

**A Timer és BT perifériát IT-sen kezeljük. A Timer 0.5sec-onént kér IT. A BT minden BT0 változásra (lenyomás/felengedés) kér IT-t. Timer IT esetén meginvertálja az LD0 LED-et majd egy N=3-as modulusú szoftver számlálót dekrementál és ha az 0, akkor jelzést ad a főprogramnak az r13-ban. A Timer IT-t a státus kiolvasása automatikusan törli. A BT IT a kiolvasott BTIF visszaírásával törli a BT IT-t. Utána ellenőrzi, hogy a BT0 lenyomása okozta-e az IT-t. Ha igen, akkor meginvertálja az LD1 LED-et. A főprogram felprogramozza a perifériákat, majd a végtelen ciklusban ellenőrzi, hogy kapott-e jelzést a Tim IT-től (R13=1). Ha igen, akkor meginvertálja az LD7 LED-et és törli a jelzést (R13=0)**

```
DEF LD  0x80
DEF TR  0x82
DEF TM  0x82
DEF TC  0x83
DEF TS  0x83
DEF BT  0x84
DEF BTIE 0x85
DEF BTIF 0x86
;7 6 5 4 3 2 1 0
;TIE TPS[2:0] - - TREP TEN (TC)
; 1       111    0 0   1      1
DEF TCINI 0b11110011
;TIT TPS[2:0] 0 TOUT TREP TEN (TS)
DEF TIT_MASK 0b10000000

DEF BT0 0x01   ;BT0 mask
DEF LD0 0x01
DEF LD1 0x02
DEF LD7 0x80
DEF N   3      ;software counter
               ;inic. value
      code
```

```
RESET:
   jmp main
ISR:mov r10, TS
   tst r10, #TIT_MASK
   jz ISR_next       ;Tim IT?
ISR_Tim:
   mov r10, LD
   xor r10, #LD0
   mov LD, r10       ;/LD0
   sub r12, #1
   jnz ISR_next
   mov r12, #N       ;sw counter tc
   mov r13, #1       ;signal to main progam
ISR_next:
   mov r10, BTIF
   mov BTIF, r10   ;clear BT IT-s
   mov r11, BT
   tst r10, r11    ;BT0 changed (BTIF0=1) and BT0 pressed (BT0=1)
   jz ISR_END    ;BT0 pressed?
ISR_BT:
   mov r10, LD
   xor r10, #LD1    ;/LD1
   mov LD, r10
ISR_END:
   rti               ;not rts!!!
```

```
main:
      mov r0, #121  ;0.5 sec signal
      mov TR, r0    ;inic. value
      mov r0, #TCINI
      mov TC, r0      ;control reg.
      mov r0, #BT0
      mov BTIE, r0  ;enable BT0 IT
      mov r0, #0
      mov LD, r0    ;LED clear
      mov r12, #N   ;IT sw. counter init
      mov r13, #0   ;signal from IT
      sti           ;global IT eng.
loop:               ;endless cycle
      tst r13, #1
      jz loop       ;signal from IT?
      cli           ;disable IT-s
      mov r0, LD
      xor r0, #LD7
      mov LD, r0      ;/LD7
      mov r13, #0     ;clear signal
      sti             ;enable IT-s
      jmp loop
```

# Digitális technika
# 12. EA vége