Hardver alapok

(VIIIBA01)

Megszakítások gyakorlati alkalmazása

Rácz György gyuriracz@iit.bme.hu

2022/23. tanév tavaszi félév

Bevezetés

- Megszakítás (ismétlés)
- Környezet mentése
- Külső megszakítás (INT)
- Portbit változás érzékelése (IOC)
- Egyéb periféria megszakítások
- Példák a megszakítás használatára
  - Puffer kezelése IT-ből (USART)
  - Időzítő megszakítás (TMR0)
  - Pergésmentesítés (INT bemenet)

Megszakítás

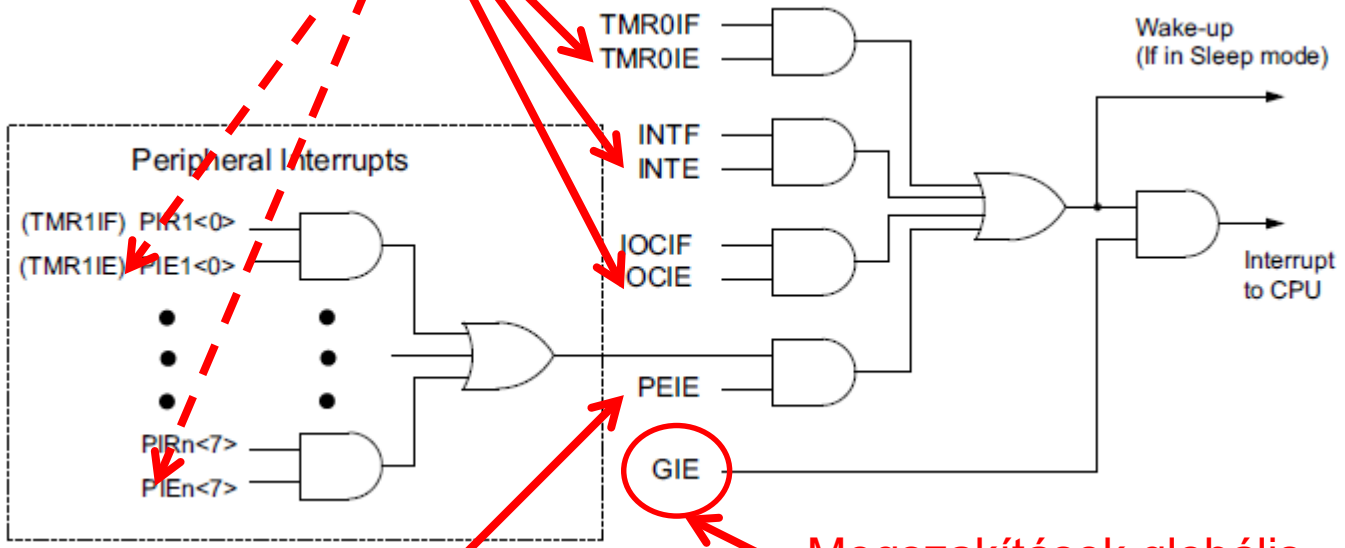
- Láttuk a szoftveres várakozó hurok és szoftveres késleltetés megvalósítását
- Mivel ez 100%-ban lefoglalja a CPU-t feleslegesen, így a CPU nem tud más tevékenységet végezni
- Megoldás: „automatikus szubrutin hívás”
- A gomb megnyomása, vagy az időzítés letelte indítson el egy rutint→megszakítási szubrutin
- IT rutin csak rövid ideig tartson

PIC16F és a megszakítás

- Sok megszakítási forrás lehet (szinte minden periféria)
- Minden forráshoz külön engedélyezés (IE) és külön jelzőbit (IF)
- Egyetlen vektor →0x0004 belépési pont
- Programból kell az okot kitalálni (melyik engedélyezett IF aktív)
- Belépéskor környezet mentése, visszatéréskor pedig a környezet visszaállítása

Megszakítás logika

Minden forrás egyedileg engedélyezhető



Perifériák külön is tilthatók

Megszakítások globális engedélyezése

REGISTER 7-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	U-0	R/W-1/1
GIE	PEIE	—	—	—	—	—	INTEDG
bit 7							bit 0

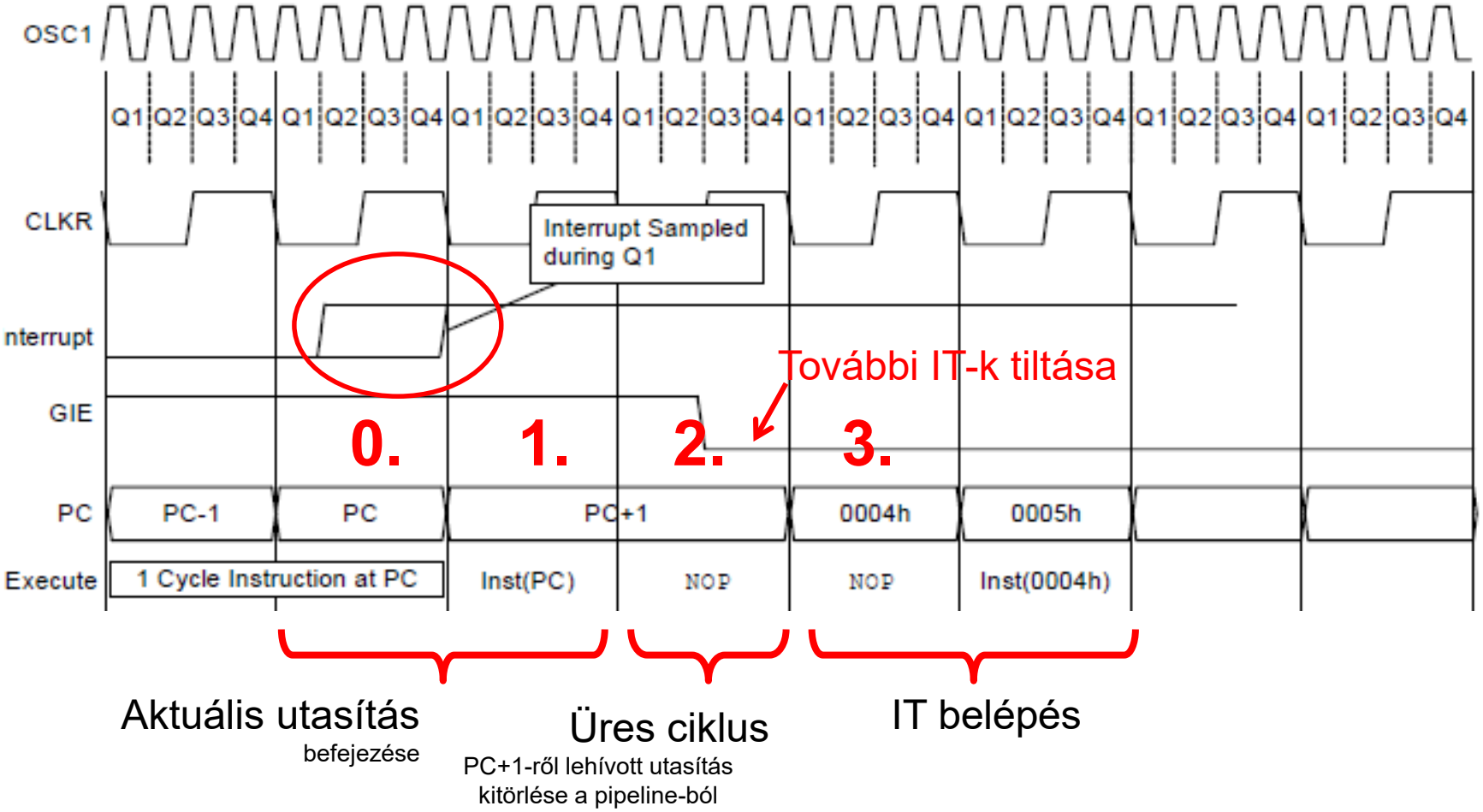
Egyedi engedélyező- és jelző bitek

PIR0…PIR8 → jelzőbitek
PIE0…PIE8 → engedélyező bitek minden perifériához
BANK14-ben érhetők el

70Ch	PIR0		—	—	TMR0IF	IOCIF	—	—	—	INTF
70Dh	PIR1		OSFIF	CSWIF	—	—	—	—	ADTIF	ADIF
70Eh	PIR2		—	ZCDIF	—	—	—	—	C2IF	C1IF
70Fh	PIR3		—	—	RCIF	TXIF	BCL2IF	SSP2IF	BCL1IF	SSP1IF
710h	PIR4		—	—	TMR6IF	TMR5IF	TMR4IF	TMR3IF	TMR2IF	TMR1IF
711h	PIR5		CLC4IF	CLC3IF	CLC2IF	CLC1IF	—	TMR5GIF	TMR3GIF	TMR1GIF
712h	PIR6		—	—	—	CCP5IF	CCP4IF	CCP3IF	CCP2IF	CCP1IF
713h	PIR7		SCANIF	CRCIF	NVMIF	NCO1IF	—	CWG3IF	CWG2IF	CWG1IF
714h	PIR8		—	—	SMT2PWAIF	SMT2PRAIF	SMT2IF	SMT1PWAIF	SMT1PRAIF	SMT1IF
715h	—	—	Unimplemented							
716h	PIE0		—	—	TMR0IE	IOCIE	—	—	—	INTE
717h	PIE1		OSFIE	CSWIE	—	—	—	—	ADTIE	ADIE
718h	PIE2		—	ZCDIE	—	—	—	—	C2IE	C1IE
719h	PIE3		—	—	RCIE	TXIE	BCL2IE	SSP2IE	BCL1IE	SSP1IE
71Ah	PIE4		—	—	TMR6IE	TMR5IE	TMR4IE	TMR3IE	TMR2IE	TMR1IE
71Bh	PIE5		CLC4IE	CLC3IE	CLC2IE	CLC1IE	—	TMR5GIE	TMR3GIE	TMR1GIE
71Ch	PIE6		—	—	—	CCP5IE	CCP4IE	CCP3IE	CCP2IE	CCP1IE
71Dh	PIE7		SCANIE	CRCIE	NVMIE	NCO1IE	—	CWG3IE	CWG2IE	CWG1IE
71Eh	PIE8		—	—	SMT2PWAIE	SMT2PRAIE	SMT2IE	SMT1PWAIE	SMT1PRAIE	SMT1IE

Megszakítás időzítése

1 ciklusú utasítás esetén



2 és 3 ciklusú utasítások esetén a lappangási idő 3..5 utasításciklus (CLKR) lehet

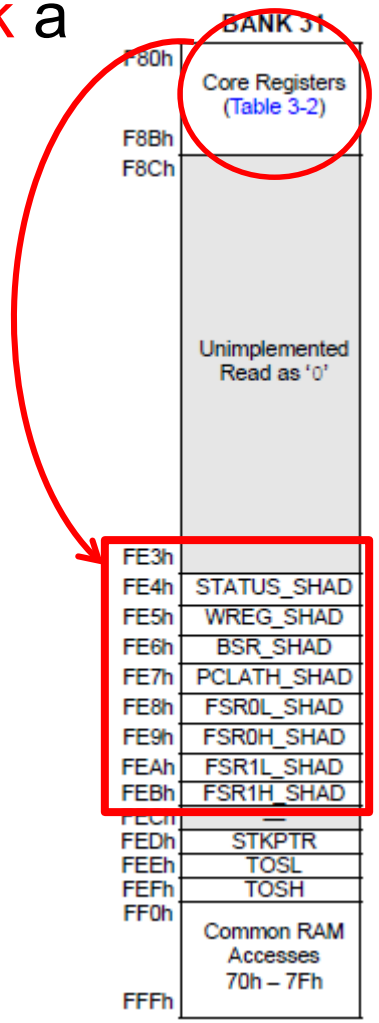
Automatikus környezet mentés

Megszakításba lépéskor automatikusan elmentődnek a RAM memória végébe (31.bank) a CORE regiszterek:

- STATUS regiszter
- W regiszter
- BSR regiszter
- PCLATH regiszter
- FSR0 és FSR1 regiszterek

Visszatéréskor az elmentett értékek helyreállításra kerülnek (RETFIE utasításra)

Minden más „rombolt” változó mentését a programozónak kell megoldania!



Külső aszinkron megszakítás bemenet

- INT bemenet (alapértelmezett: RB0)
- Bármelyik portbitre konfigurálható →INTPPS
- Programozható, hogy le vagy felfutó élre okozzon IT-t
- Kapcsolódó regiszterek:

REGISTER 7-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	U-0	R/W-1/1
GIE	PEIE	—	—	—	—	—	INTEDG
bit 7							bit 0

REGISTER 7-2: PIE0: PERIPHERAL INTERRUPT ENABLE REGISTER 0

U-0	U-0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
—	—	TMR0IE	IOCIE	—	—	—	INTE
bit 7							bit 0

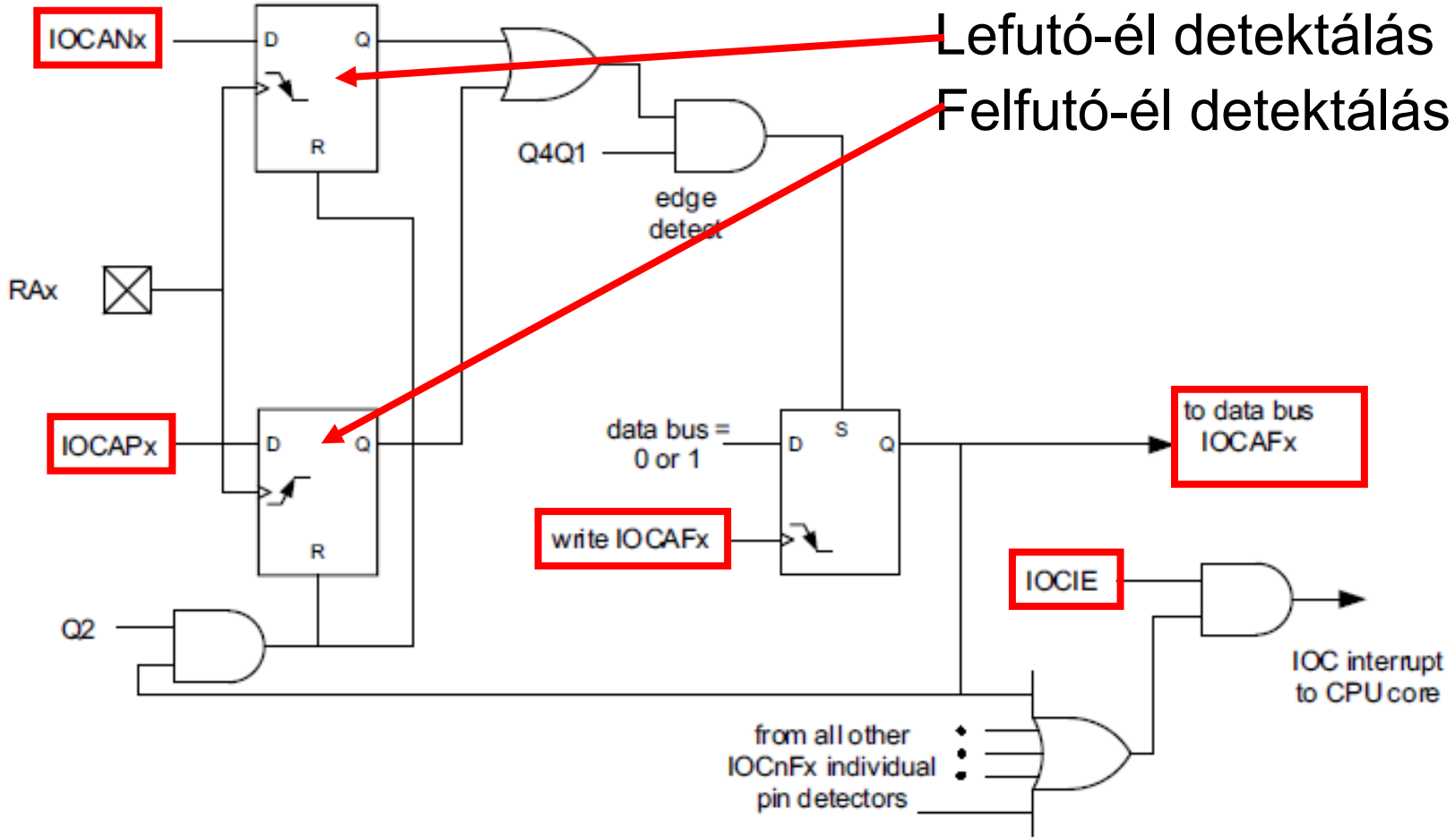
REGISTER 7-11: PIR0: PERIPHERAL INTERRUPT STATUS REGISTER 0

U-0	U-0	R/W/HS-0/0	R-0	U-0	U-0	U-0	R/W/HS-0/0
—	—	TMR0IF	IOCIF	—	—	—	INTF(1)
bit 7							bit 0

Portbit változás érzékelése

- IOC – Interrupt On Change
- Szinte mindegyik portbit megváltozása okozhat megszakítást
- Portbitenként állítható, hogy felfutó, lefutó vagy mindkét élre
- Pontos forrás (melyik láb változott) kiderítése már program feladat
- Az egész modul megszakításkérése egyben tiltható →IOCIE=0

I/O változás érzékelése



Lefutó-él detektálás

Felfutó-él detektálás

IOC regiszterek (PORTA)

REGISTER 15-1: IOCAP: INTERRUPT-ON-CHANGE PORTA POSITIVE EDGE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IOCAP7	IOCAP6	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0
bit 7							bit 0

IOCAPi=1→felfutó él érzékelés engedélyezett

REGISTER 15-2: IOCAN: INTERRUPT-ON-CHANGE PORTA NEGATIVE EDGE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IOCAN7	IOCAN6	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0
bit 7							bit 0

IOCANi=1→lefutó él érzékelés engedélyezett

REGISTER 15-3: IOCAF: INTERRUPT-ON-CHANGE PORTA FLAG REGISTER

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
IOCAF7	IOCAF6	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0
bit 7							bit 0

IOCAFi=1→megérkezett a beállított élváltás

Ugyanígy tartoznak regiszterek a PORTB, PORTC-hez is.

IOC megszakítás nyugtázása

- 0 értéket kell írni a megszakítást okozó IOCxF regiszter kiszolgált bitjére.
- Ha a törlés elmarad, nem szűnik meg az IT kérés
- Ha nincs szükség a bitek megkülönböztetésére, akkor 0x00-t írhatunk az IOCxF regiszterbe

IT rutin előkészítése

- Programmemória 0x0004 címén belépési pont beállítása
- Környezet mentése
- Engedélyezett IT-k vizsgálata
- IT kérések kiszolgálása
- Környezet visszaállítás
- Visszatérés+IT engedélyezés

Belépési pont beállítása

```
#include "p16f18875.inc"
 __CONFIG _CONFIG1, 0x3F8C
 __CONFIG _CONFIG2, 0x3F3F
 __CONFIG _CONFIG3, 0x3F9F
 __CONFIG _CONFIG4, 0x3FFF
 __CONFIG _CONFIG5, 0x3FFF
; Reset Vector
RES_VECT    CODE    0x0000 ; processzor reset vektor
    GOTO    START   ; ugrás a főprogram elejére
ISR_VECT    CODE    0x0004 ; megszakítás vektor
;     . . . . ; innentől jöhet az IT rutin kódja
MAIN_PROG   CODE   ; főprogram helye a kódszegmensben
START:
;     . . . . ; innentől jöhet a főprogram
             END
```

Változó terület definiálása (RAM)

```
GPR_VAR      UDATA H'020' ; BANK0, változó terület
DBCOUNT      RES 1 ; darabszám
MSCNT        RES 1 ; IT, osztó
SEC          RES 1 ; IT, másodpercek
MIN          RES 1 ; IT, percek
HOUR         RES 1 ; IT, órák

UARTBUFF     UDATA H’0A0’ ; BANK1, GPR kezdete
TXBUFF       RES 16 ; helyfoglalás az adó puffernek
```

IT okának kiderítése

- Pl.: TMR0 és UART TX esemény

```
ISR_VECT    CODE    0x0004 ; megszakítás vektor
   BANKSEL     PIE0
   MOVF        PIE0,W ; engedélyező bitek
   ANDWF       PIR0,W ; flag bitek ÉS kapcsolata
   BTFSC       WREG,TMR0IF
   GOTO        TMR0_IT       ; TMR0 IT rutin

   MOVF        PIE3,W ; engedélyező bitek
   ANDWF       PIR3,W ; flag bitek ÉS kapcsolata
   BTFSC       WREG,TXIF
   GOTO        SEND_IT       ; USART TX rutin

ISR_V:
    RETFIE
```

UART periféria kezelés példa

Készítsünk egy szubrutint, amely 16 bájtot továbbít a TXBUFF tömbből USART-on.

(9600BPS, 8adatbit, nincs paritás)

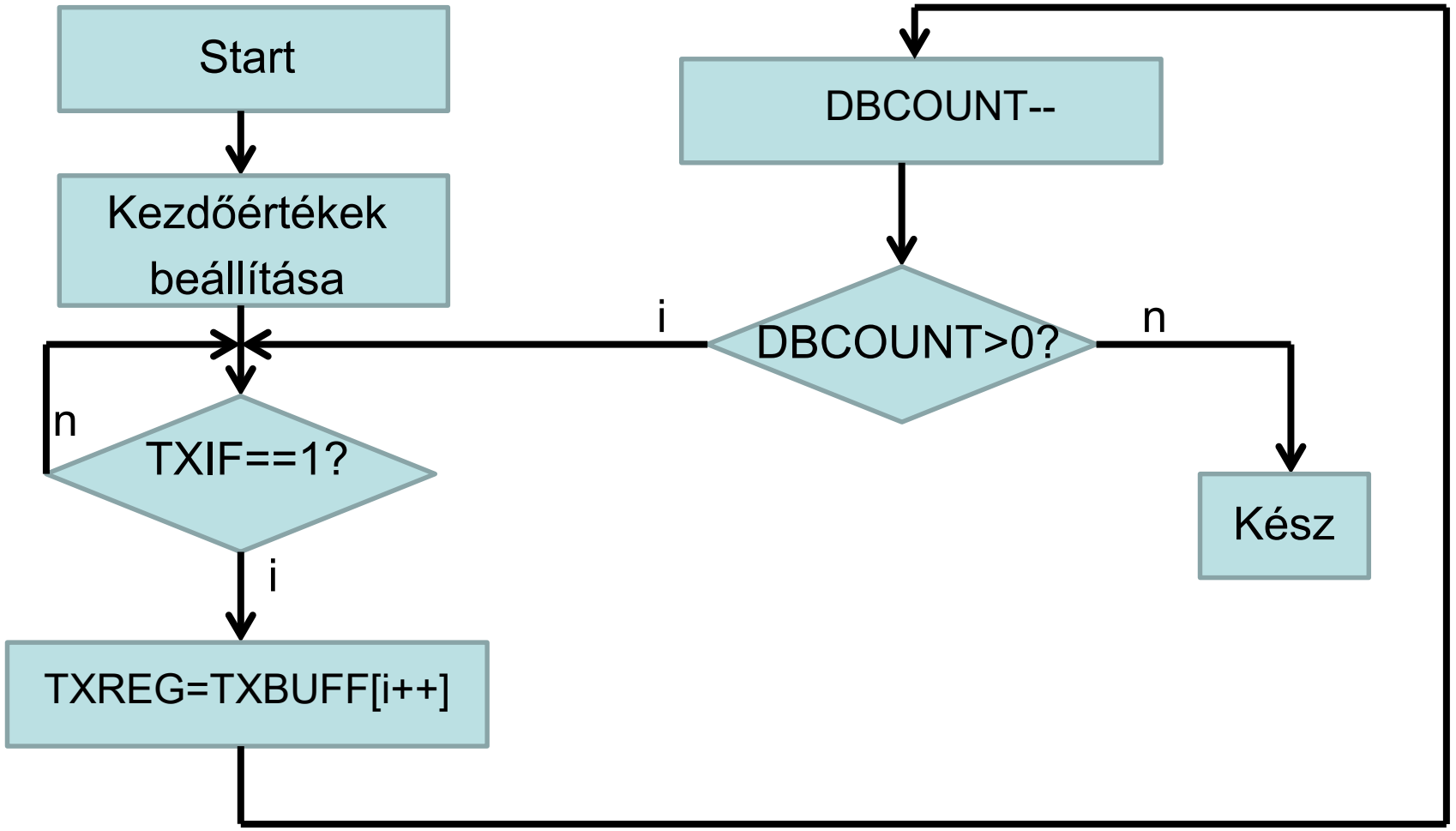
A) programmal ellenőrzött készenléttel

B) Megszakításból

(tételezzük fel, hogy az UART már inicializált állapotban van, nem foglalkozunk bankváltásokkal.)

Határozzuk meg a CPU kihasználtságot a fenti két esetben

SW készenlét



Megoldás SW készenléttel

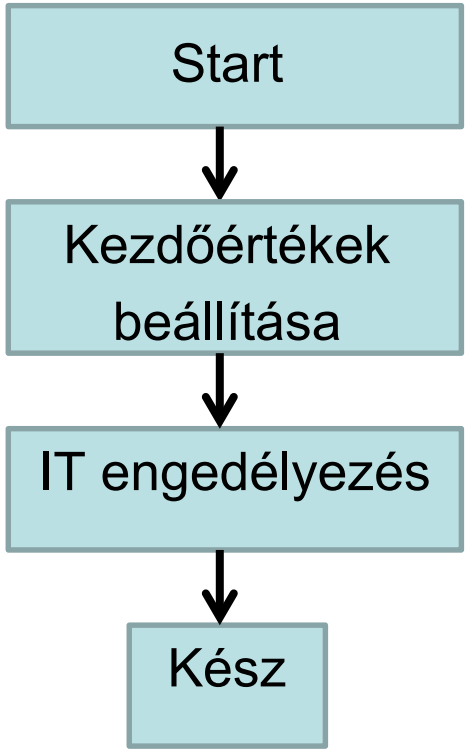
```
SEND_BUFF:
   MOVLW     HIGH TXBUFF ; mutató beállítása
   MOVWF     FSR0H
   MOVLW     LOW TXBUFF  ;két részletben (16bites)
   MOVWF     FSR0L
   MOVLW     D'16'       ; darab számláló
   MOVWF     DBCOUNT
SND_CK:
   BTFSS     PIR3,TXIF   ;üres adóra vár
   BRA       SND_CK
   MOVIW     FSR0++      ; adat W-be, pointer ++
   MOVWF     TXREG       ; beírás az adóba
   DECFSZ    DBCOUNT,F
   BRA       SND_CK
   RETURN
```

Futási idő becslése

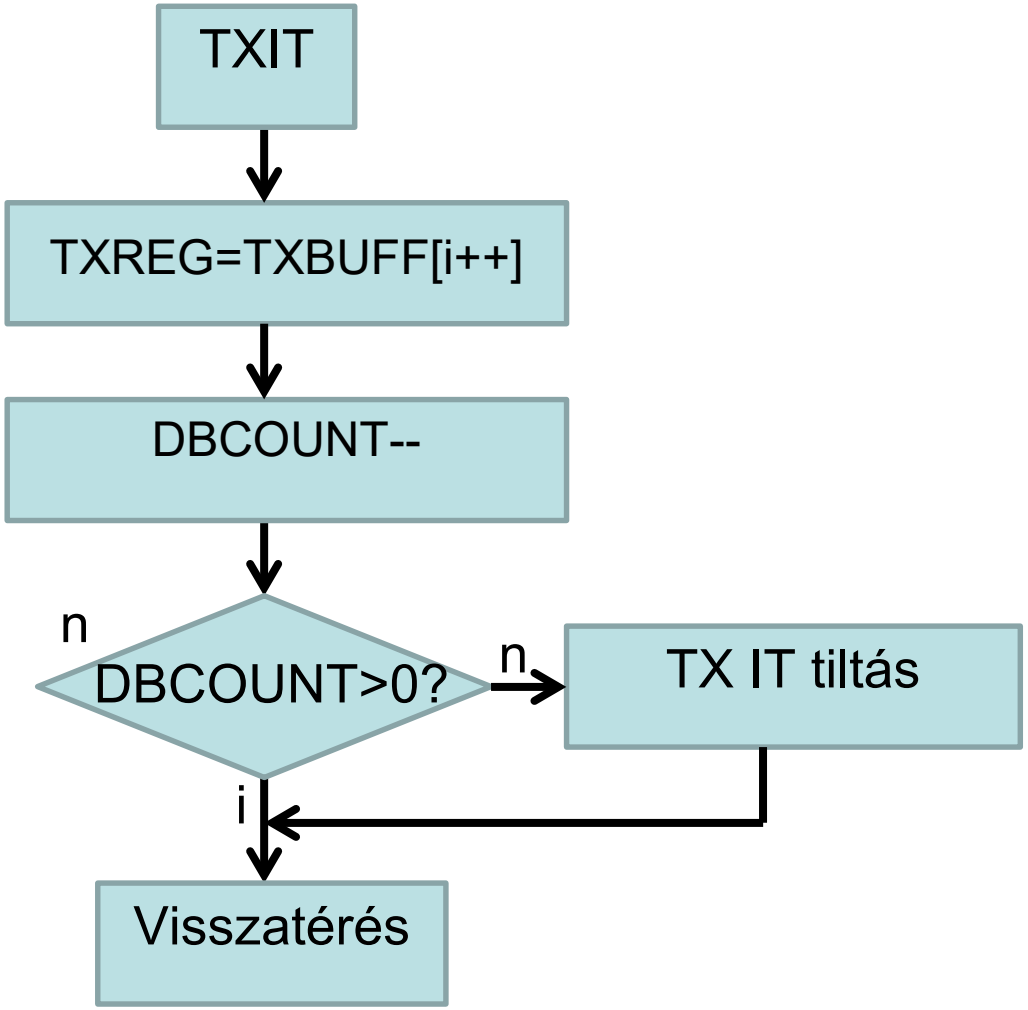
- Előkészítés, visszatérés: 8 ciklus→1us
- Egy iteráció, feltéve, hogy nem kell az usart-ra várni: 7 ciklus→0,875us
- 1 karakter elküldése (10 bit, 9600bps)→1041,6us
- Mivel bufferelt az adó→15 karakternyi időt kell várakozással tölteni (az utolsó már kimegy)→
  Tbecsült ≅ 1+1+15·(1+1041,6)=15.641ms
- A teljes elküldés 16·1041,6=16.6656ms ideig tart
- Ezalatt a CPU 93.85%-ban kihasznált, az idő nagy részében vár az UART-ra.

Küldés IT-ből

Főprogram



Megszakítás



Megoldás IT-vel

```
    ORG 0x0004
SEND_IT:
  BTFSS      PIE3,TXIE ; IT engedélyezve?
  BRA IT_V
  BTFSS      PIR3,TXIF ; adó üres?
  BRA IT_V
  BANKSEL    TXREG
  MOVIW      FSR0++ ; küldendő karakter W-be, pointer ++
  MOVWF      TXREG  ; beírás az adóba
  DECF       DBCOUNT,F
  BANKSEL    PIE3
  BTFSC      STATUS,Z
  BCF        PIE3,TXIE     ; további IT tiltása
  BANKSEL    FSR0L_SHAD
  MOVF       FSR0L,W
  MOVWF      FSR0L_SHAD
  MOVF       FSR0H,W
  MOVWF      FSR0H_SHAD
IT_V:
  RETFIE
```

Nem kell, ha máshol már megvizsgáltuk…

Az automatikus visszaállítás ne állítsa vissza a megnövelt pointert…

Megoldás IT-vel

```
SEND_BUFF:
   MOVLW          HIGH TXBUFF   ; mutató beállítása
   MOVWF          FSR0H
   MOVLW          LOW TXBUFF
   MOVWF          FSR0L
   MOVLW          D'16'         ; darab számláló
   MOVWF          DBCOUNT
   BANKSEL        PIE3
   BSF        PIE3,TXIE
   BSF        INTCON, PEIE
   BSF        INTCON, GIE
   RETURN
```

Futási idő becslése

- Indító szubrutin: 11ciklus→1.375us
- IT lappangás: max 5 ciklus→0,625us
- IT lefutása: 18 ciklus→2,25us
- A teljes CPU ciklus igény (11+16x(18+5))ciklus→47.375us
- A teljes elküldés 16·1041,6=16.6656ms ideig tart
- Ezalatt a CPU 0,28%-ban kihasznált, az idő nagy részében mást is csinálhat.

Időzítő megszakítás példa

Készítsünk valós idejű órát (RTC-t) a TMR0 időzítő megszakítás felhasználásával.

A program az órákat, perceket, másodperceket a HOUR, MIN, SEC változókban tárolja.

Ötlet a megoldáshoz

- Használjuk 8 bites módban a TMR0-t, így pontosan beállíthatjuk az IT kérés periódusidejét→10ms
- Szoftveres utóosztás 1s-hez (/100)
- Szoftveres utóosztás 1p-hez (/60)
- Szoftveres utóosztás 1h-hoz (/60)

IT kérés TMR0-al

Példa: kérjünk 10ms-időnként IT-t a TMR0-al

$$100Hz = 8000000Hz \cdot \frac{1}{32} \cdot \frac{1}{250} \cdot \frac{1}{10}$$

Regiszter név	Érték (binárisan)	Megjegyzés
T0CON0	B’1000 1001’	Bekapcsolás + 1:10 utóosztás
T0CON1	B’0100 0101’	Fosc/4 + 1:32 előosztás
TMR0H	B’1111 1001’	0…249 között számol

REGISTER 27-1: T0CON0: TIMER0 CONTROL REGISTER 0

R/W-0/0	U-0	R-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
T0EN	—	T0OUT	T016BIT	T0OUTPS<3:0>			
bit 7							bit 0

REGISTER 27-2: T0CON1: TIMER0 CONTROL REGISTER 1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
T0CS<2:0>			T0ASYNC	T0CKPS<3:0>			
bit 7							bit 0

Inicializálás

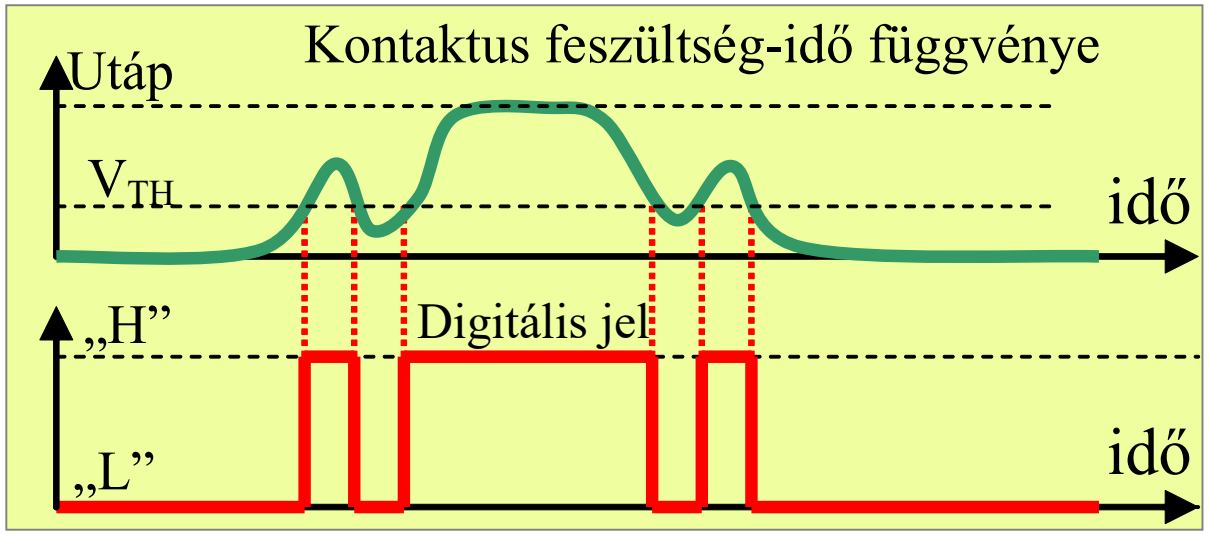
```
TMR0INIT:
   MOVLW      B'10001001'
   MOVWF      T0CON0
   MOVLW      B'01000101'
   MOVWF      T0CON1
   MOVLW      D'249'
   MOVWF      TMR0H
   MOVLW      D’100’ ; 1sec, SW utóosztás
   MOVWF      MSCNT
   CLRF       SEC    ; kezdőérték
   CLRF       MIN
   CLRF       HOUR
   BANKSEL    PIR0   ; BANK14
   BCF        PIR0, TMR0IF
   BSF        PIE0, TMR0IE
   BSF        INTCON,GIE
   RETURN
```

IT rutin

```
TMR0IT:
    BANKSEL     PIR0
    BTFSS       PIR0,TMR0IF
    BRA         TMRIT_V
    BCF         PIR0, TMR0IF
    DECFSZ      MSCNT,F ; 10ms
    BRA         TMRIT_V
    MOVLW       D'100' ; 1sec
    MOVWF       MSCNT
    INCF        SEC,F
    MOVF        SEC,W
    SUBLW       D'60'
    BTFSS       STATUS,Z
    BRA         TMRIT_V
    CLRF        SEC
    INCF        MIN,F   ; 1perc
    MOVF        MIN,W
    SUBLW       D'60'   ; 60-W
    BTFSS       STATUS,Z
    BRA         TMRIT_V
    CLRF        MIN
    INCF        HOUR,F  ; 1óra
    MOVF        HOUR,W
    SUBLW       D'24'   ; 24-W
    BTFSS       STATUS,Z
    BRA         TMRIT_V
    CLRF        HOUR
TMRIT_V:
    RETFIE
```

Pergésmentesítés IT-ből

- Ha külső mechanikus nyomógombbal szeretnénk IT-t kérni, egy lenyomásra több IT is jöhet



Lehetséges megoldások:

- Ha egy IT jött, néhány ms időre tiltjuk
- Másik megoldás: a gomb nem okoz it-t, hanem periodikusan mintavételezzük egy időzítő megszakításból

IT rutin

```
INT_IT:
   BANKSEL     PIR0
   BCF         PIR0, INTF
   BCF         PIE0, INTE
   BANKSEL     T0CON0
   BSF         T0CON0,T0EN
   BSF         GOMBSTAT,0; Jelzés
   BRA IT_V
TMR0_IT:
   BANKSEL     T0CON0
   BCF         T0CON0,T0EN
   BANKSEL     PIR0
   BCF         PIR0,TMR0IF
   BCF         PIR0,INTF
   BSF         PIE0,INTE
   BRA IT_V
IT_V:
   RETFIE
```

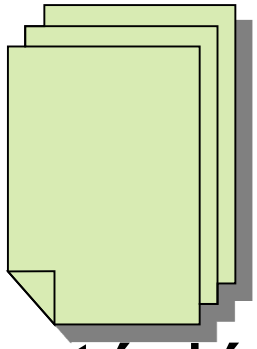
```
Init_IT:
  BCF  GOMBSTAT,0
  BANKSEL     T0CON0
  MOVLW       B‘00001001'
  MOVWF       T0CON0
  MOVLW       B'01000101'
  MOVWF       T0CON1
  MOVLW       D'249'
  MOVWF       TMR0H
  CLRF TMR0L
  BANKSEL     PIR0
  BCF  PIR0,TMR0IF
  BCF  PIR0,INTF
  BSF  PIE0,TMR0IE
  BSF  PIE0,INTE
  BSF  INTCON,PEIE
  BSF  INTCON,GIE
  RETURN
```

Megoldás értékelése

Problémák:

- Csak 1db INT megszakítás van → lehetne az IOC-t haszálni
- Nem célszerű minden gombhoz külön HW időzítőt használni
- Sok gomb esetén célszerűbb az időzítő megszakításból olvasni a nyomógomb bemeneteket

A témához kapcsolódó anyagok



- Microchip weboldala

  http://www.microchip.com
- MPLABX fejlesztői környezet és dokumentáció letöltése

  https://www.microchip.com/mplab/mplab-x-ide
- Online help a mikrokontrollerhez és a fejlesztői környezethez

  http://microchipdeveloper.com/mcu1102:start
- Tárgy honlapján

  https://www.iit.bme.hu