Realizace zpoždění

Instrukce trvá většinou 1 instrukční cyklus ($4 \times T_{CLK}$), skoky 2

Časová smyčka – nejkratší trvá 3 cykly, maximální 256×3+2

```
        movlw   POCET   ; počet průběhů -> W
        movwf   citac   ; naplnění registru - čítače průběhů
Loop:   decfsz  citac,F ; dekrementace a test na nulu
        goto    Loop    ; skok na opakování smyčky
        ...             ; při nule přeskok – pokračování prog.
        ...
        movlw   POCET   ; počet průběhů -> W
        decfsz  WREG,W  ; dekrementace a test na nulu
        goto    $-1     ; skok na opakování smyčky
        ...             ; při nule přeskok – pokračování prog.
```

Lze doladit vložením několika instrukcí NOP do těla smyčky nebo na konec

Vícenásobná smyčka

```
; hlavní program
            ...
            movlw   NMS         ; zadání počtu milisekund
            call    Delay_ms    ; volání podprogramu zpoždění NMS ms
            ...                 ; přesně N * 1000 + 5 cyklů

Delay_ms:   movwf   cnt2        ; naplnění čítače vnější smyčky z W
OutLp:      movlw   249         ; trvání smyčky 249*4+4=1000 cyklů
            movwf   cnt1        ; naplnění čítače vnitřní smyčky
InLp:       nop                 ; tělo vnitřní smyčky
            decfsz  cnt1,F
            goto    InLp        ; skok na opakování vnitřní smyčky
            decfsz  cnt2,F
            goto    OutLp       ; skok na opakování vnější smyčky
            return
```

Při f_{clk} = 4 MHz (T_{instr} = 1 μs) čeká podprogram N milisekund
Parametr N se předává v registru W
Možno vložit více smyček

Čekání na úroveň

Reakce na asynchronní událost

```
Loop:   btfsc  i,j     ; test j-tého bitu registru i na 0
        ;(btfss  i,j     ; test j-tého bitu registru i na 1)
        goto   Loop    ; skok na opakování smyčky
        ...            ; při splnění podmínky – pokračování
```

- vhodné doplnit na začátku test jiného bitu, aby bylo možné čekání přerušit
- žádné instrukce skoků v programu neovlivňují příznaky

Čekání na hranu

- test obou úrovní za sebou

Příkl.: čekání na sestupnou hranu na čtrvtém bitu portu A

```
LoopH:  btfss   PORTA,4  ; test na úroveň H (log.1)
        goto    LoopH
LoopL:  btfsc   PORTA,4  ; test na úroveň L (log.0)
        goto    LoopL
        ...              ; při splnění podmínky – pokračování


        btfss   PORTA,4  ; test na úroveň H (log. 1)
        goto    $-1
        btfsc   PORTA,4  ; test na úroveň L (log. 0)
        goto    $-1
        ...              ; při splnění podmínky – pokračování
```

Adresování – přímé a nepřímé

Adresování paměti dat

- *přímé* (7-bitová konstanta „k“ je součástí instrukce)
- *nepřímé* (ukazatelem jsou FSR a obsah je v INDF)