EURÓPSKA ÚNIA

Program celoživotného vzdelávania

Programovanie PIC procesora

Vypracoval: Ing. Jaroslav Janoušek

Tento edukačný materiál vznikol v rámci projektu Programu celoživotného vzdelávania
Leonardo da Vinci – č. 11323 1208
„Európske skúsenosti a zručnosti v elektrotechnických školách“

Vzorová hodina na tému

„Ako programovať PIC + vzorový príklad“

Najprv sa zoznámime s mikropočítačom

Mikropočítače PIC sú programovateľné polovodičové súčiastky (jedno čipové mikropočítače) celý počítač je integrovaný do jedného čipu (viď. bloková schéma príloha 3) vyrábané firmou Microchip Technology. Stačí zapojiť príslušné vstupy a výstupy, napájanie prípadne taktovacie impulzy a zložité zapojenie s mnohými súčiastkami za nás urobí príslušný program.

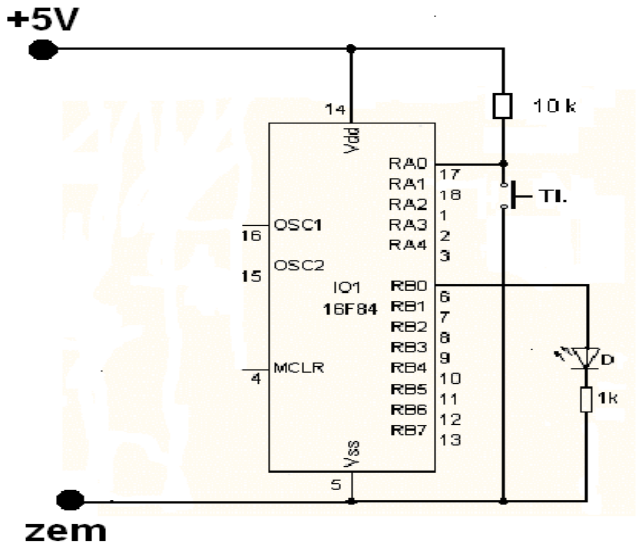
Každý procesor má dve alebo tri brány s príslušným počtom pinov. Každý z týchto pinov môžeme používať buď ako vstupné, alebo ako výstupné. Pokiaľ je používaný ako vstupní(RA0) tak má veľký vstupný odpor, preto musíme odporom R definovať logickú 1 (+5V). U niektorých procesorov sa tento odpor dá nastaviť programovo (fyzicky to nie je potrebné). Log. 0 získame pripojením tohto vstupu na zem. Pokiaľ PIN príslušnej brány (RB0) nastavíme na výstup potom program na tento PIN posiela logickú 0 (0V), resp. logickú 1 (5V). Toto napätie priamo môžeme priviesť na LED diódu, zosilniť tranzistorom, nebo ním spínať relé. Nastavení pinu ako vstupný, alebo výstupný vykonáme programovo (register TRISA, TRISB, …).

Ako každý procesor i PIC potrebuje hodinový impulzy. Ten môžeme vytvoriť RC článkom, u niektorého procesora (627) môžeme použiť interní oscilátor. V prípade, že potrebujeme presnú frekvenciu oscilátora na vstupy OSC1 a OSC2, pripojíme kryštál (kondenzátor, ktorý umožní kmitať len na jeho prípustné hodnoty). Z toho môžeme odvodiť doby časových cyklov.

Vlastnosti procesorov sú popísané v prílohe 2.

Tieto programovateľné súčiastky sú určené pre najrôznejšie kontrolné a riadiace úlohy v priemyselných oboroch, pre realizáciu meracích a riadiacich systémov atď. Vďaka svojej univerzálnosti, malej veľkosti, nízkej cene a spotrebe nachádzajú uplatnenie vo veľkom množstve aplikácií. Môžete ich nájsť v mnohých súčasných elektronických zariadeniach.

Zapojení vstupu a výstupu mikroprocesorov.



Nasleduje ukážka programovania mikroprocesorov PIC.

Každý mikroprocesor má svoju inštrukčnú sadu (viď. prílohy 6 a 7).
Inštrukcie procesoru v strojovom kóde (nazývanej aj binárny kód) sú zvyčajne uložené v hexadecimálnom tvare (súbor s príponou.hex). V tomto tvare by sa nám program veľmi ťažko program písal, preto sa miesto strojového kódu v hexadecimálnom tvare programuje v Asembleri, v ktorom sú inštrukcie v zrozumiteľnom tvare.

Program sa píše v zvyčajne v textovom editore napr. poznámkovom bloku a ukladá sa v súbore s koncovkou **.ASM**. Inštrukcie sa píšu do troch stĺpcov, ktoré sa oddeľujú znakom **TAB**. S výnimkou príkazov pre definovanie názvov sa do prvého stĺpca píšu návestia, do druhého príkazy a do tretieho stĺpca ich hodnoty. Všetko musíme písať veľkými písmenami.

Každý správny programátor, by mal do programu vkladať poznámky (ich účelom je hlavne zrozumiteľnosť pre cudzích programátorov, po čase aj autora, ktorý na detaily zabudne). Poznámky sa píšu za bodkočiarku"**;**" a malými písmenami.

Písanie programu najlepšie ozrejmí príklad.

Chceme vytvoriť program pre mikropočítačov, ktorý po stlačení tlačidla rozsvieti LED diódu a pri nasledovnom stlačení tlačidla LED diódu zhasne. Použijeme už uvedenú schému zapojenia.

Program musí začínať Hlavičkou, v ktorej sa uvedie typ procesora a pridelenie knižnice

```
LIST P=16F84                ; určuje pre aký typ procesora
INCLUDE<P16F84.INC>         ; priradenie knižnice procesora
```

Ďalej musíme nakonfigurovať procesor, kde nastavíme jeho vlastnosti.

Pre úplnosť uvádzam v prílohe konfiguráciu 16F627A (príloha 8)

V príklade pre jednoduchosť uvádzam 16F84. Konfiguráciu ku každému procesoru nájdete v Datashetu.

Konfiguráciu nemusíme písať pokiaľ vie nakonfigurovať procesor vývojové prostredie, alebo programátor.

V ukážke uvádzam spozdenie po štarte je zapnuté, Watchdog timer je vypnutý, typ oscilátoru RC

```
CONFIG_PWRTE_ON&_WDT_OFF&_RC_OSC ;konfigurácia procesora
```

Aby sme nemuseli písať v programu PORTA, 0 a podobne môžeme nadefinovať promennú napríklad TLAC.

```
#DEFINE    TLAC    PORTA, 0 ; nastavenie názvov používaných vývodov neskôr sa v
                              ;programe môže používať už len tieto názvy a nie ich adresy
```

Podobne definujeme výstup premennej LED.

```
#DEFINE    LED     PORTB, 0
```

Teraz musíme prepnúť príslušné piny na vstupné, alebo výstupné, toto uskutočníme v registroch TRISA,TRISB. Nastavením na hodnotu log.1 sú vstupné, nastavením na log.0 sú výstupné. Skôr než budeme nastavovať musíme prepnúť do Banku 1.

```
BSF     STATUS, RP 0  ; nastavení vývodov vstupní/výstupní, musíme
                      ; sa prepnúť do Banky 1

                      ; To sa stalo týmto príkazom, ktorý nám bit RP0
                      ; v registru STATUS, nastavil na hodnotu log.1
```

Teraz nastavíme bit 1 v registru TRISA na hodnotu log.1-prvý bit vstupný, ostatné výstupné. Informáciu najskôr pošleme do W registra (základný register vstupu/výstupu) a následne pošleme do TRISA, TRISB môžeme celý vynulovať, teda prepnúť na výstupný-

```
MOVLW   B'00000001' ; uložení binárnej hodnoty do pracovného registra
MOVWF   TRISA       ;prepísanie z pracovného registra do registra TRISA

                    ; nastavení vývodu RA0 ako vstupné/ostatné
                    ;vývody ako výstupné.

CLRF    TRISB       ;do TRISB zapíšeme samé 0 a tým celý prepneme
                    ;ako výstupný
```

Následne nazad prepneme register do Bank 0. A môžeme ho využiť ako vstupný, alebo výstupný.

```
BCF     STATUS, RP0 ; po nastavení sa vrátime späť do Banky 0 opäť
                    ; zmenou bitu RP0 tentoraz však na log.0
```

Po tomto máme nastavené porty a môžeme začať vlastný program:

Bit nastavíme na nulu. Nultý bit portu B (definovali sme mu premennú LED).

```
BCF   LED           ; na log. 0 sme nastavili vývod s názvom LED
```

Aby sme mohli vytvoriť programový cyklus, použijeme návästie START.

Teraz budeme testovať stlačenie tlačidla inštrukcia BTFSS má skokovú funkciu, preto za ňu musíme dať inštrukciu GOTO $-1, tým čakáme na stlačenie tlačidla. Po stlačení tlačidla preskočí inštrukciu GOTO $-1 a program pokračuje ďalej.

```
START   BTFSS   TLAC ; Slovo START slúži ako  záložka (pre neskoršie
                     ; použitie stačí GOTO START
                     ; Príkaz za záložkou testuje vývod označený TLAC
                     ; Z tabuľky príkazov je vidieť, že ak je tento vývod v 0
                     ; pokračuje sa normálne ďalším príkazom.
                     ; Ak má hodnotu 1, nasledujúci riadok sa vynechá a
                     ; pokračuje sa až tým ďalším
                     ;(v našom prípade je to BSF LED).
        GOTO    $-1  ; príkaz spolu s jeho hodnotou spôsobí vrátenie procesora o
                     ; jeden riadok vyššie ($-1)
                     ; Spolu s minulým príkazom sme dosiahli to, že
                     ; procesor tu stále opakuje cyklus a čaká,
                     ; až bude mať vývod TLAC hodnotu 1
                     ; V tej chvíli sa tento príkaz preskočí a procesor sa dostane
                     ; na nasledovný riadok
```

Následne chceme rozsvietiť LED diódu, preto na premennú LED nastavíme príslušný bit portu B na log.1

```
BSF     LED ; nastavení hodnoty 1 na vývod LED a tým rozsvietenie
                ;pripojenej  LED diódy
```

Teraz čakáme na pustenie tlačidla, preto použijeme podobný cyklus ako v predchádzajúcom prípade, ale s obrátenou funkciou.

```
BSFSC   TLAC        ; musíme priviesť na vývod TLAC log.0
        GOTO    $-1 ; vrátenie procesoru o jeden riadok vyššie
```

Teraz budeme opäť testovať stlačenie tlačidla.

```
        BTFSS   TLAC ; rovnaký princíp až na to, že sa tlačidlom LED vypne
        GOTO    $-1
```

Následne chceme zhasnúť LED diódu, preto na premennú LED nastavíme príslušný bit portu B na log.0.

```
        BSF     LED
```

Opäť testujeme tlačidlo.

```
        BTFSC   TLAC
        GOTO    $-1
```

Aby sme vytvorili cyklus, vraciame sa na návestie START.

```
        GOTO    START ; procesor sa vráti späť na záložku START a tým
                      ; vytvoríme opakujúci sa cyklus rozsvietenia a zhasnutia LED
```

Každý program je zakončený príkazom END pre prekladač.

```
        END ;  indikuje koniec programu, musí sa písať! (Napriek tomu, že program
                ; na tento príkaz nikdy nedôjde)
```

Tu uvádzam skrátenú verziu celého programu:

```
;Hlavičky programu
LIST P=16F84                                    ; typ procesoru
INCLUDE<P16F84.INC>                             ; knižnice
CONFIG_PWRTE_ON&_WDT_OFF&_RC_OSC  ;určenie vlastností
#DEFINE         TLAC    PORTA, 0
#DEFINE         LED             PORTA, 1

                 BSF     STATUS, RP 0 ;
                MOVLW    B'00000001'
                MOVWF   TRISA
                BCF          STATUS, RP0
                BCF         LED

START           BTFSS     TLAC
                GOTO      $-1
                BSF         LED
                BSFSC     TLAC
                GOTO      $-1
                BTFSS     TLAC
                GOTO      $-1
                BSF         LED
                BTFSC     TLAC
                GOTO      $-1
                GOTO      START
END
```

Takto napísaný program uložíme s koncovkou .asm

Pomocou prekladača - napríklad MPASMWIN, preložíme náš súbor do strojového kódu . Pokiaľ preklad prebehne správne, vznikne nám súbor s príponou .HEX.
Teraz už stačí k PC pripojiť programátor a vygenerovaný súbor naprogramovať do procesora a celé zapojenie odskúšať.

Prekladač nám ukáže iba syntaktické chyby, ale logické chyby v programe neukáže. Nie je teda isté, že takto napísaný program bude fungovať.

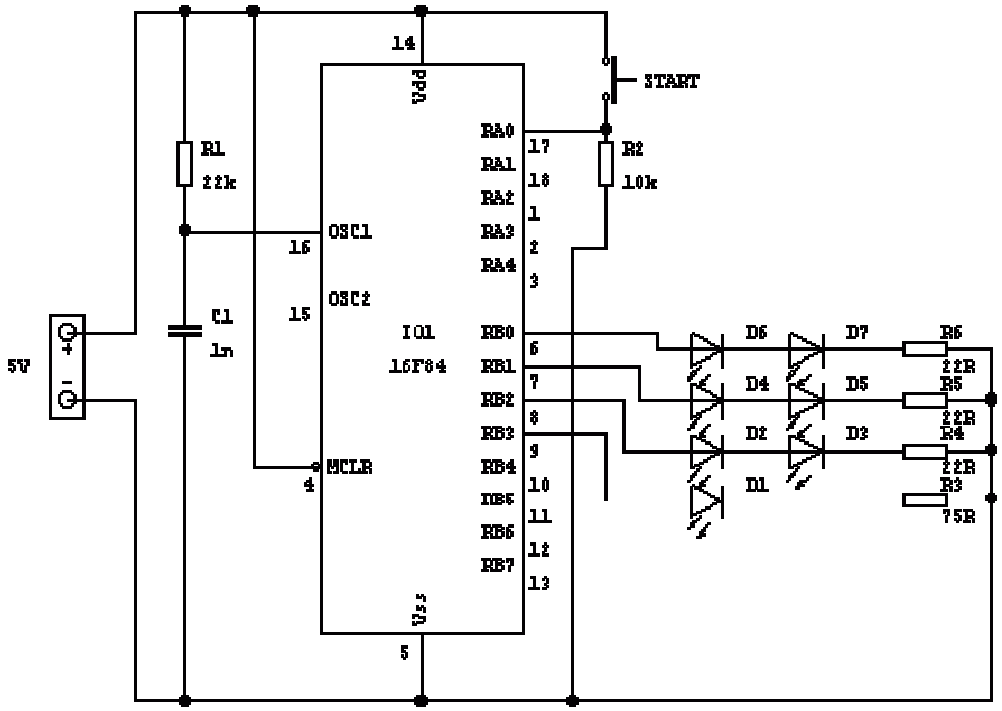
K odladeniu programu je preto vhodné použiť nejaké vývojové prostredie, kde môžeme nasimulovať funkciu programu. Ako jeden z vývojových programov je vhodný MPLAB.

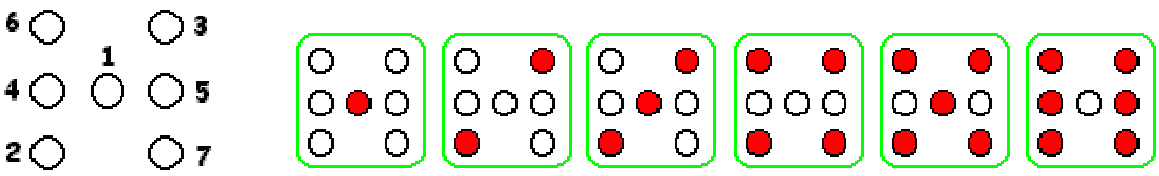
Vzorová aplikácia

Človeče nehnevaj sa! s PIC

Ide o elektronickú verziu klasickej hracej kocky používanej hlavne na hru Človeče nehnevaj sa.

Schéma zapojenia je na obr. Tlačidlo je pripojené na vývod RA0. Na vývodoch RB0 až RB3 sú proti zemi pripojené LED diódy, ktorých rozmiestnenie je vidieť na obr. 2.
Ako oscilátor je tu použitý obyčajný RC člen (nie je potrebná presná frekvencia) a pracuje asi na 45 kHz.

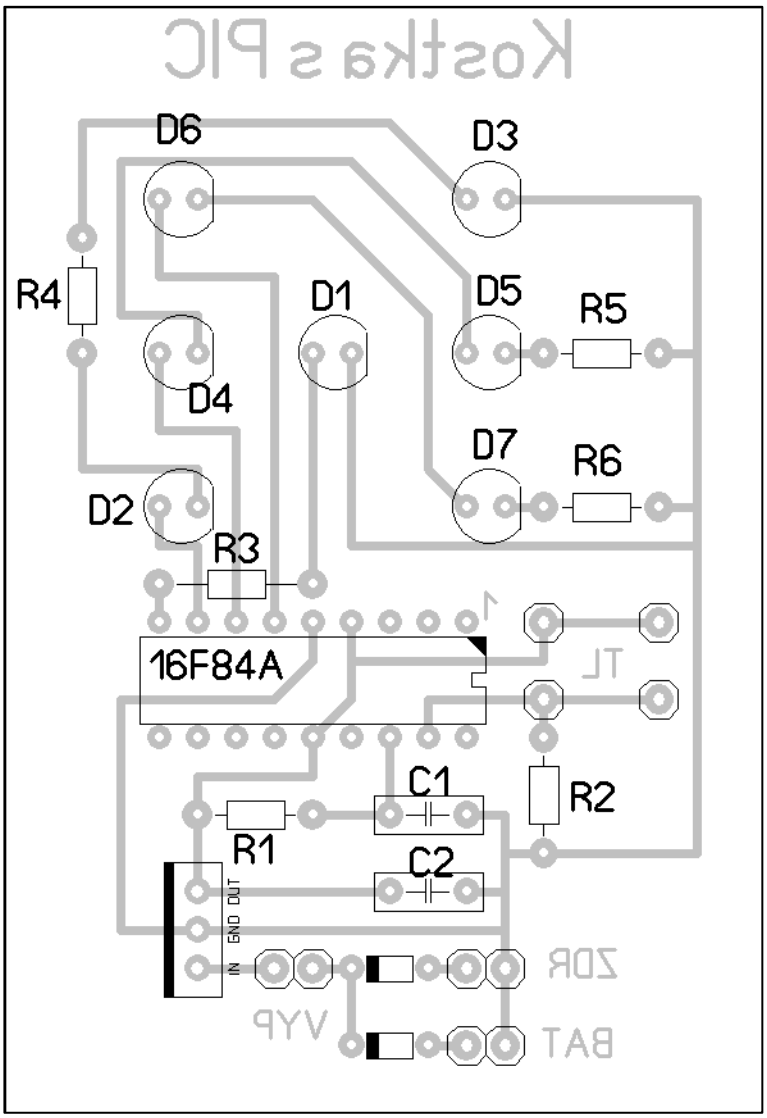


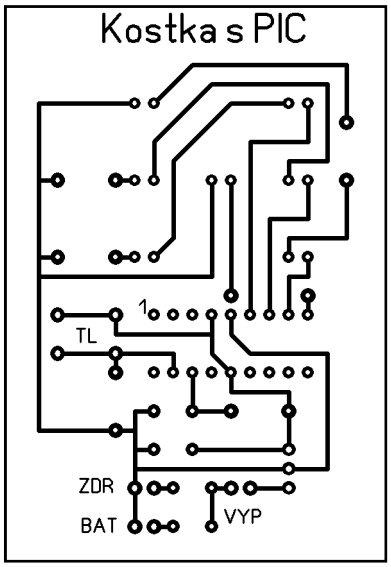


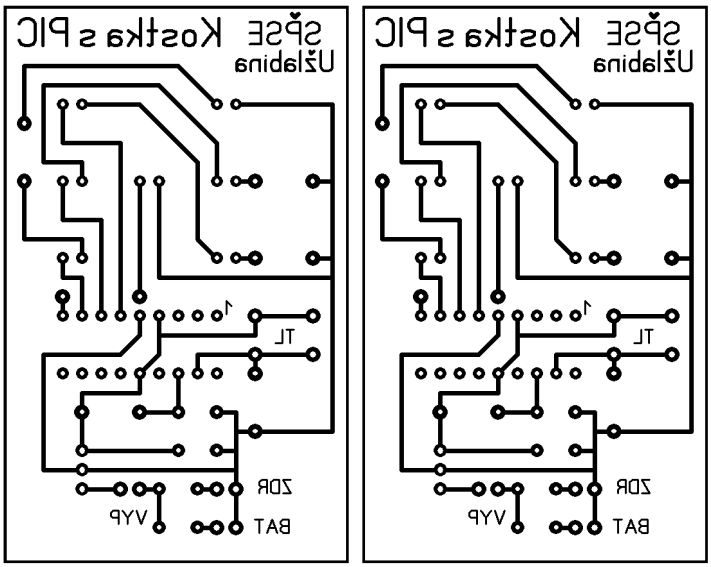
Človeče nehnevaj sa! s PIC

súčiastky a plošné spoje

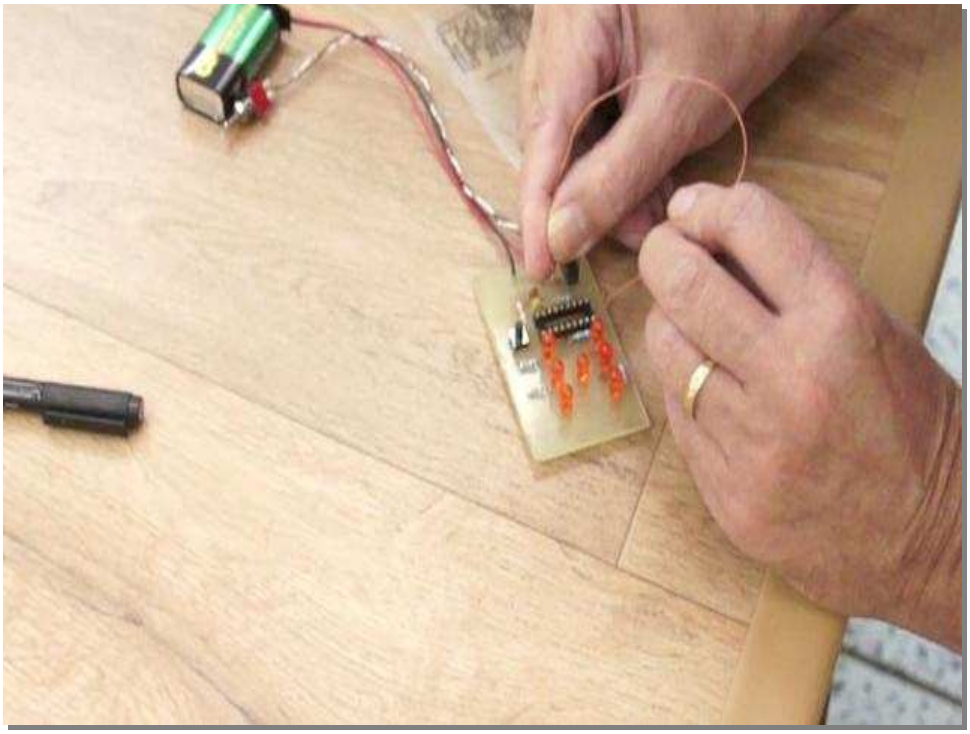
R1 - 22k
R2 - 10k
R3 - 75R
R4 - 22R
R5 - 22R
R6 - 22R
C1 - 1n
C2 - 100n
D1 – D6 Led dióda 20mA







Oživovanie elektronickej kocky



```
; Elektronická kocka určená pre hru Človeče nehnevaj sa!, vhodná aj pre iné hry
;
;
; Na RA0 je tlačidlo START prepojené voči plusu a na porte B sú LED.
; Po stlačení tlačidla START sa rozsvietia všetky LED
; a procesor "mieša" jednotlivé čísla.
; Po uvoľnení sa aktuálna hodnota zapíše na PORT B.
;

        LIST P=16F84, R=DEC     ; typ procesora a numerická dekadická sústava
        INCLUDE<P16F84.INC>     ; vzorový súbor s príkazmi pre prekladač (je
súčasťou
                               ;  prekladača ASM/HEX)

                                ; Všetky tu používané registre majú už svoje
označenia
                                ; a nie je ich potrebné tu nastavovať.
                                 ; Napr.: PORTA, PORTB,STATUS ...

        BSF     STATUS,RP0      ; vyber v registroch BANK 1
        MOVLW   B'00000001'     ; do W zapíš tuto hodnotu
        MOVWF   TRISA           ; hodnotu z W zapíš do TRISA a tým nastav prvý
vývod na IN
        CLRF    TRISB           ; do TRISB zapíš samé 0 a tým cely PORTB nastav
ako OUT
        BCF     STATUS,5        ; vyber naspäť BANK 0
        MOVLW   B'00001000'     ; toto číslo ulož do W
        MOVWF   PORTB           ; W do PORTB

START   BTFSS   PORTA,0         ; čaká na stlačenie tlačidla START na vývode RA0
        GOTO    START           ; pokiaľ je stlačené, tento príkaz sa preskočí
inak nazad
        MOVLW   B'00001111'     ; binárne číslo do W
        MOVWF   PORTB           ; W do PORTB - rozsvietia sa všetky LED

MICH    MOVLW   B'00001000'     ; do W hodnotu pre prvé číslo
        BTFSS   PORTA,0         ; ak je ešte stlačene tl, nasledujúci riadok sa
vynechá
        GOTO    ZOBRAZ          ; prejdi na ZOBRAZ

        MOVLW   B'00000100'     ; a opäť pre ďalšie číslo
        BTFSS   PORTA,0
        GOTO    ZOBRAZ

        MOVLW   B'00001100'
        BTFSS   PORTA,0
        GOTO    ZOBRAZ

        MOVLW   B'00000101'
        BTFSS   PORTA,0
        GOTO    ZOBRAZ

        MOVLW   B'00001101'
        BTFSS   PORTA,0
        GOTO    ZOBRAZ

        MOVLW   B'00000111'
        BTFSS   PORTA,0
        GOTO    ZOBRAZ
        GOTO    MICH            ; a opäť miešať

ZOBRAZ  MOVWF   PORTB           ; hodnotu z W zapíš na PORTB
        GOTO    START           ; prejdi na START

        END
```