Dekadická korekce

- používá se při výpočtech ve dvojkově-desítkové soustavě (číslice jsou nejčastěji v BCD kódu – kód 8421)
- pokud při sčítání je nižší nibble větší než 9, přičteme 6; obdobně při odečítání od mezivýsledku odečteme 6, je-li nibble větší než 9

Př.: 26 + 35 = 61

```
  0010 0110
  0011 0101
-------------
  0101 1011  (= 5Bh)
  0000 0110  (+ 6)
-------------
  0110 0001  (= 61)
```

71 – 56 = 15

```
  0111 0001
- 0101 0110
-------------
  0001 1011  (= 1Bh)
- 0000 0110  (- 6)
-------------
  0001 0101  (= 15)
```

Převod z binární soustavy do BCD

Celé nezáporné binární číslo *N* lze vyjádřit:

$$N = a_n.2^n + a_{n-1}.2^{n-1} + \ldots + a_1.2^1 + a_0$$

$$N = ((a_n.2 + a_{n-1}).2 + \ldots + a_1).2 + a_0$$

Číslo *N* lze vytvořit postupným násobením dvěma (posun vlevo) a přičítáním následujícího bitu (0 nebo 1). Díváme se na číslo jako dekadické, používáme dekadickou korekci.

Někdy se používá metoda postupného odečítání mocnin základu (doba převodu závisí se převáděném čísle) nebo metoda postupného dělení základem - hodnotou 10 (relativně zdlouhavé).

Převod z binární soustavy na BCD

Převeďte binární číslo 0110100 ($a_6 \ldots a_0$) na číslo dekadické v BCD kódu

0000 0000	počáteční stav
0000 000**0**	přičtení a_6
0000 000**0**	~~dekadická korekce~~
0000 00**01**	$a_6.2 + a_5$
0000 00**01**	~~dekadická korekce~~
0000 0**011**	$(a_6.2 + a_5).2 + a_4$
0000 0**011**	~~dekadická korekce~~
0000 **0110**	$((a_6.2 + a_5).2 + a_4).2 + a_3$
0000 **0110**	~~dekadická korekce~~
000**0 1101**	$(((a_6.2 + a_5).2 + a_4).2 + a_3).2 + a_2$
000**1 0011**	dekadická korekce
00**10 0110**	$((((a_6.2 + a_5).2 + a_4).2 + a_3).2 + a_2).2 + a_1$
00**10 0110**	~~dekadická korekce~~
0**100 1100**	$(((((a_6.2 + a_5).2 + a_4).2 + a_3).2 + a_2).2 + a_1).2 + a_0$
0**101 0010**	dekadická korekce … **výsledek 52**

Příklad

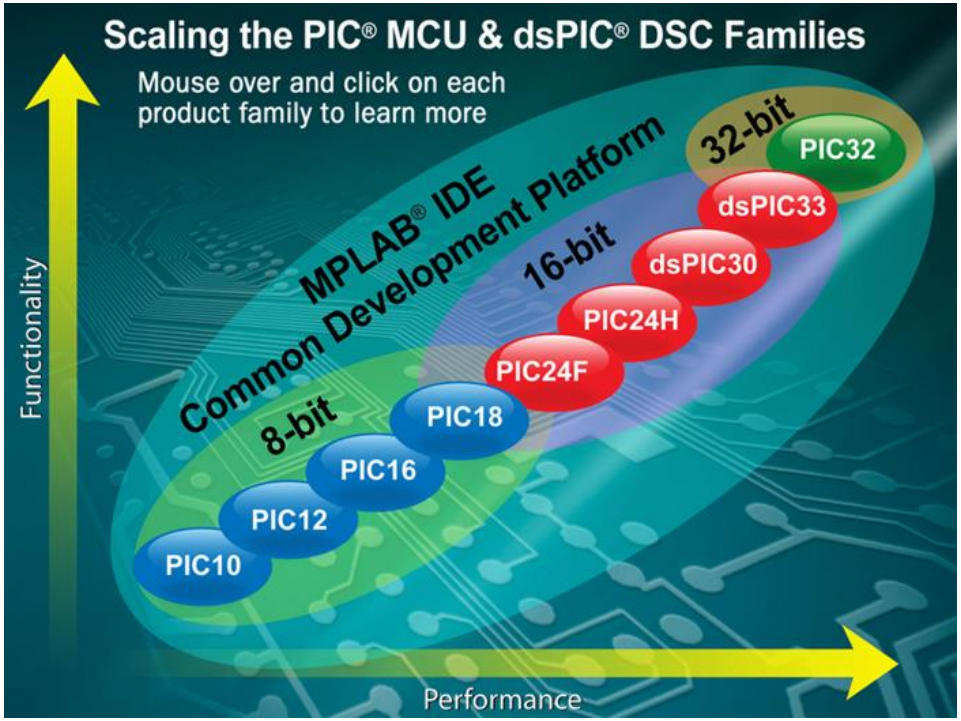
Sečtěte $(154)_{10}$ a $(271)_{10}$ v BCD kódu.

Příklad

Sečtěte $(154)_{10}$ a $(271)_{10}$ v BCD kódu.

$(154)_{10}$ =	0001	0101	0100		
$+(271)_{10}$ =	0010	0111	0001		
	0011	1100	0101		
	+	0110			
	0100	0010	0101	=	$(425)_{10}$

Mikrořadiče PIC



Mikrořadiče Microchip PIC16

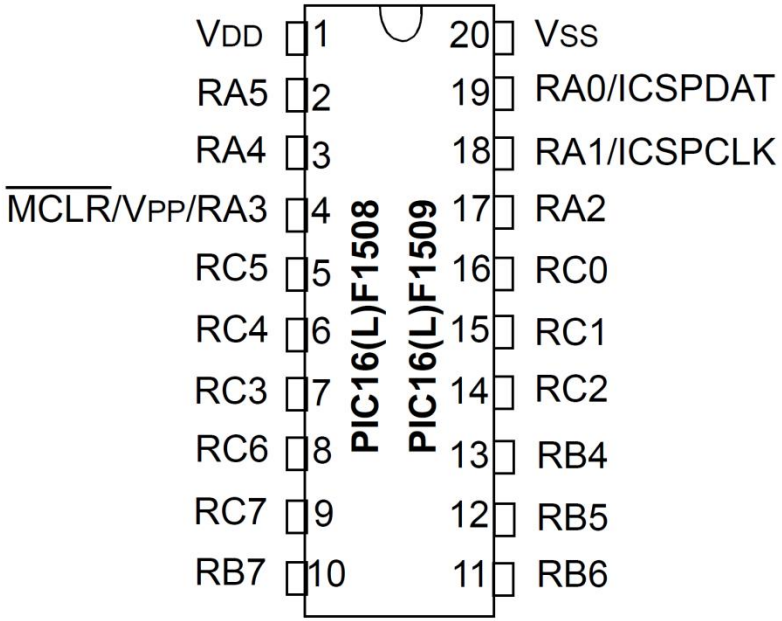
- RISC mikrořadiče v technologii CMOS

- harvardská arch.: programová sběrnice 14 bitů, datová 8 bitů
- všechny instrukce jsou jedno-cyklové (kromě skokových - podle výsledku operace jsou jedno- až dvou-cyklové)
- jeden strojový takt trvá 4 hodinové pulsy
- dvoustupňový pipelining (fetch, execute)

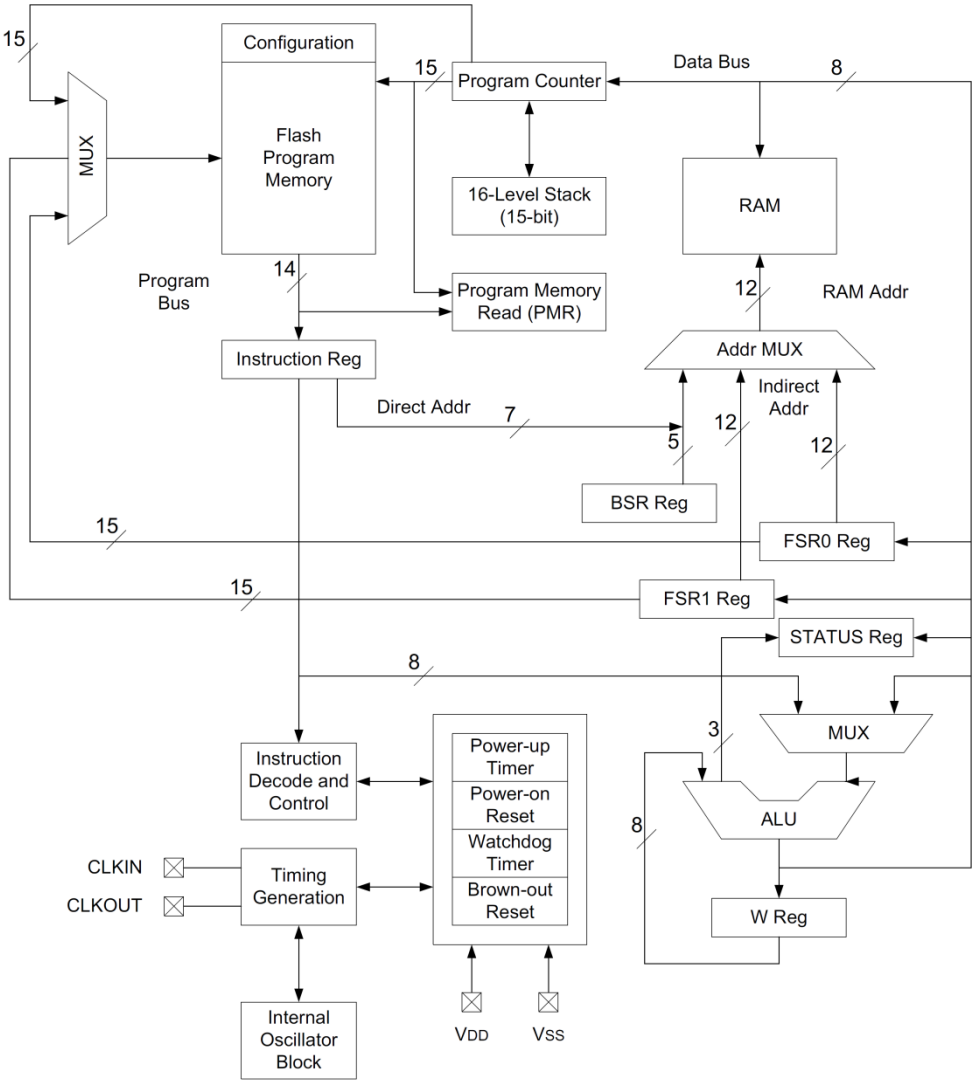
- velmi nízká proudová spotřeba
- rychlost je až 12 MIPS (48 MHz)

Mikrořadič PIC16F1508

- pouzdro s 20 vývody
- 4096 slov programové paměti - FLASH
- 16-úrovňový zásobník
- 256 B Data RAM
- 128 B Data HEF (EEPROM)
- max. hod. kmitočet 20 MHz
- 200 ns min. instr. cyklus
- napájení 2,3 – 5,5 V
- ~1,1 mA @ 5 V @ 16 MHz
- programování ICSP po 5 pinech
- odběr z výstupů až ±25 mA



Jádro a paměti PIC16F1508



Watchdog Timer (WDT)

Hlídací časovač – obvod zajišťující kontrolu správného běhu programu (musí být vynulován/nastaven dříve než dojde k jeho přetečení)

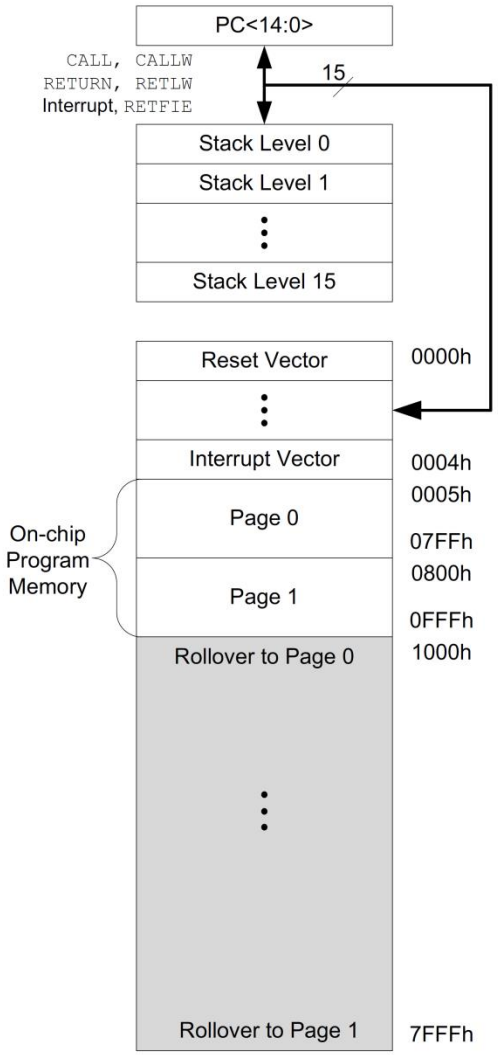
- pokud není periodicky resetován, při přetečení provede reset mikrořadiče
- používá hlavní hodinový signál nebo nezávislý interní RC oscilátor
- doba plného načítání obvykle volitelná
  – zde časy od 1 ms do 256 s (kroky po mocninách 2)
- slouží pro vzpamatování aplikace po zběhnutí programu

Programová paměť

Program Counter – viz dále

Stack – viz dále

Programová paměť – FLASH
- velikost 32k × 14 bitů
  (0000h až 7FFFh)
- min. 10 000 přepisů
  (min. 100 000 pro HEF)



Stack (zásobník)

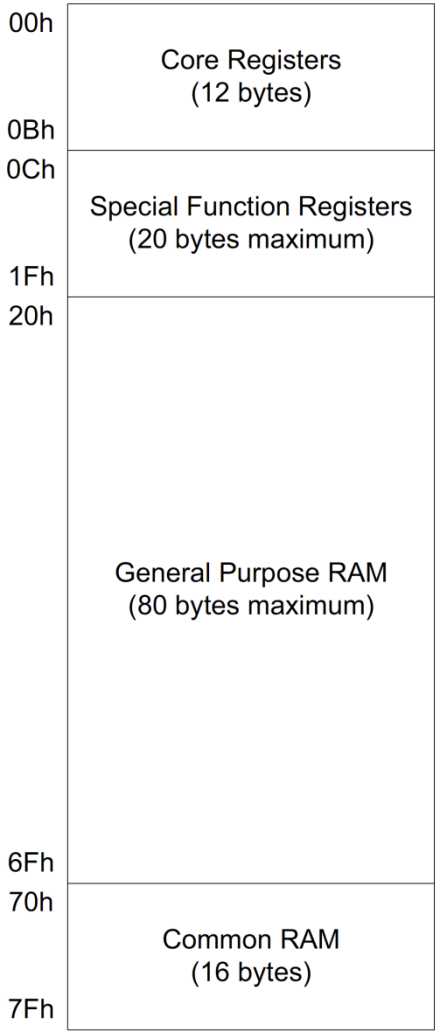
- 16 úrovní × 15 bitů (je to adresní zásobník)

- ukládá návratovou adresu při skoku do podprogramu
  (CALL, *CALLW* nebo přerušení*)*
- obnovuje návratovou adresu při návratu z podprogramu
  (RETURN, *RETLW* nebo *RETFIE)*

- není součástí adresového prostoru pro program nebo data
- SP (Stack Pointer, ukazatel zásobníku) nelze SW ovládat
  (neexistují zde instrukce *PUSH* a *POP*)
- přetečení/podtečení ošetřeno – *RESET* mikrokontroleru

Paměť RAM

- rozdělena na 32 bank
- každá banka rozdělena na:
  - 12 core registers (kopie napříč bankami)
  - až 20 SFR (dle periferií)
  - až 80 B volné RAM (vždy různých)
  - 16 B volné RAM (kopie napříč bankami)



Paměť RAM – banky 0-7

BANK 0	BANK 1	BANK 2	BANK 3	BANK 4	BANK 5	BANK 6	BANK 7
000h – 00Bh: Core Registers (Table 3-2)	080h – 08Bh: Core Registers (Table 3-2)	100h – 10Bh: Core Registers (Table 3-2)	180h – 18Bh: Core Registers (Table 3-2)	200h – 20Bh: Core Registers (Table 3-2)	280h – 28Bh: Core Registers (Table 3-2)	300h – 30Bh: Core Registers (Table 3-2)	380h – 38Bh: Core Registers (Table 3-2)
00Ch PORTA	08Ch TRISA	10Ch LATA	18Ch ANSELA	20Ch WPUA	28Ch —	30Ch —	38Ch —
00Dh PORTB	08Dh TRISB	10Dh LATB	18Dh ANSELB	20Dh WPUB	28Dh —	30Dh —	38Dh —
00Eh PORTC	08Eh TRISC	10Eh LATC	18Eh ANSELC	20Eh —	28Eh —	30Eh —	38Eh —
00Fh —	08Fh —	10Fh —	18Fh —	20Fh —	28Fh —	30Fh —	38Fh —
010h —	090h —	110h —	190h —	210h —	290h —	310h —	390h —
011h PIR1	091h PIE1	111h CM1CON0	191h PMADRL	211h SSP1BUF	291h —	311h —	391h IOCAP
012h PIR2	092h PIE2	112h CM1CON1	192h PMADRH	212h SSP1ADD	292h —	312h —	392h IOCAN
013h PIR3	093h PIE3	113h CM2CON0	193h PMDATL	213h SSP1MSK	293h —	313h —	393h IOCAF
014h —	094h —	114h CM2CON1	194h PMDATH	214h SSP1STAT	294h —	314h —	394h IOCBP
015h TMR0	095h OPTION_REG	115h CMOUT	195h PMCON1	215h SSP1CON1	295h —	315h —	395h IOCBN
016h TMR1L	096h PCON	116h BORCON	196h PMCON2	216h SSP1CON2	296h —	316h —	396h IOCBF
017h TMR1H	097h WDTCON	117h FVRCON	197h VREGCON	217h SSP1CON3	297h —	317h —	397h —
018h T1CON	098h —	118h DAC1CON0	198h —	218h —	298h —	318h —	398h —
019h T1GCON	099h OSCCON	119h DAC1CON1	199h RCREG	219h —	299h —	319h —	399h —
01Ah TMR2	09Ah OSCSTAT	11Ah —	19Ah TXREG	21Ah —	29Ah —	31Ah —	39Ah —
01Bh PR2	09Bh ADRESL	11Bh —	19Bh SPBRG	21Bh —	29Bh —	31Bh —	39Bh —
01Ch T2CON	09Ch ADRESH	11Ch —	19Ch SPBRGH	21Ch —	29Ch —	31Ch —	39Ch —
01Dh —	09Dh ADCON0	11Dh APFCON	19Dh RCSTA	21Dh —	29Dh —	31Dh —	39Dh —
01Eh —	09Eh ADCON1	11Eh —	19Eh TXSTA	21Eh —	29Eh —	31Eh —	39Eh —
01Fh —	09Fh ADCON2	11Fh —	19Fh BAUDCON	21Fh —	29Fh —	31Fh —	39Fh —
020h – 06Fh: General Purpose Register 80 Bytes	0A0h – 0EFh: General Purpose Register 80 Bytes	120h – 16Fh: General Purpose Register 80 Bytes	1A0h – 1EFh: Unimplemented Read as '0'	220h – 26Fh: Unimplemented Read as '0'	2A0h – 2EFh: Unimplemented Read as '0'	320h – 36Fh: Unimplemented Read as '0'	3A0h – 3EFh: Unimplemented Read as '0'
070h – 07Fh: Common RAM	0F0h – 0FFh: Common RAM (Accesses 70h – 7Fh)	170h – 17Fh: Common RAM (Accesses 70h – 7Fh)	1F0h – 1FFh: Common RAM (Accesses 70h – 7Fh)	270h – 27Fh: Common RAM (Accesses 70h – 7Fh)	2F0h – 2FFh: Common RAM (Accesses 70h – 7Fh)	370h – 37Fh: Common RAM (Accesses 70h – 7Fh)	3F0h – 3FFh: Common RAM (Accesses 70h – 7Fh)

Legend: = Unimplemented data memory locations, read as '0'.

Paměť RAM – banky 8-29

BANK 8		BANK 9		BANK 10		BANK 11		BANK 12		BANK 13		BANK 14		BANK 15	
400h	Core Registers (Table 3-2)	480h	Core Registers (Table 3-2)	500h	Core Registers (Table 3-2)	580h	Core Registers (Table 3-2)	600h	Core Registers (Table 3-2)	680h	Core Registers (Table 3-2)	700h	Core Registers (Table 3-2)	780h	Core Registers (Table 3-2)
40Bh		48Bh		50Bh		58Bh		60Bh		68Bh		70Bh		78Bh	
40Ch	—	48Ch	—	50Ch	—	58Ch	—	60Ch	—	68Ch	—	70Ch	—	78Ch	—
40Dh	—	48Dh	—	50Dh	—	58Dh	—	60Dh	—	68Dh	—	70Dh	—	78Dh	—
40Eh	—	48Eh	—	50Eh	—	58Eh	—	60Eh	—	68Eh	—	70Eh	—	78Eh	—
40Fh	—	48Fh	—	50Fh	—	58Fh	—	60Fh	—	68Fh	—	70Fh	—	78Fh	—
410h	—	490h	—	510h	—	590h	—	610h	—	690h	—	710h	—	790h	—
411h	—	491h	—	511h	—	591h	—	611h	PWM1DCL	691h	CWG1DBR	711h	—	791h	—
412h	—	492h	—	512h	—	592h	—	612h	PWM1DCH	692h	CWG1DBF	712h	—	792h	—
413h	—	493h	—	513h	—	593h	—	613h	PWM1CON	693h	CWG1CON0	713h	—	793h	—
414h	—	494h	—	514h	—	594h	—	614h	PWM2DCL	694h	CWG1CON1	714h	—	794h	—
415h	—	495h	—	515h	—	595h	—	615h	PWM2DCH	695h	CWG1CON2	715h	—	795h	—
416h	—	496h	—	516h	—	596h	—	616h	PWM2CON	696h	—	716h	—	796h	—
417h	—	497h	—	517h	—	597h	—	617h	PWM3DCL	697h	—	717h	—	797h	—
418h	—	498h	NCO1ACCL	518h	—	598h	—	618h	PWM3DCH	698h	—	718h	—	798h	—
419h	—	499h	NCO1ACCH	519h	—	599h	—	619h	PWM3CON	699h	—	719h	—	799h	—
41Ah	—	49Ah	NCO1ACCU	51Ah	—	59Ah	—	61Ah	PWM4DCL	69Ah	—	71Ah	—	79Ah	—
41Bh	—	49Bh	NCO1INCL	51Bh	—	59Bh	—	61Bh	PWM4DCH	69Bh	—	71Bh	—	79Bh	—
41Ch	—	49Ch	NCO1INCH	51Ch	—	59Ch	—	61Ch	PWM4CON	69Ch	—	71Ch	—	79Ch	—
41Dh	—	49Dh	—	51Dh	—	59Dh	—	61Dh	—	69Dh	—	71Dh	—	79Dh	—
41Eh	—	49Eh	NCO1CON	51Eh	—	59Eh	—	61Eh	—	69Eh	—	71Eh	—	79Eh	—
41Fh	—	49Fh	NCO1CLK	51Fh	—	59Fh	—	61Fh	—	69Fh	—	71Fh	—	79Fh	—
420h	Unimplemented Read as '0'	4A0h	Unimplemented Read as '0'	520h	Unimplemented Read as '0'	5A0h	Unimplemented Read as '0'	620h	Unimplemented Read as '0'	6A0h	Unimplemented Read as '0'	720h	Unimplemented Read as '0'	7A0h	Unimplemented Read as '0'
46Fh		4EFh		56Fh		5EFh		66Fh		6EFh		76Fh		7EFh	
470h	Accesses 70h – 7Fh	4F0h	Accesses 70h – 7Fh	570h	Accesses 70h – 7Fh	5F0h	Accesses 70h – 7Fh	670h	Accesses 70h – 7Fh	6F0h	Accesses 70h – 7Fh	770h	Accesses 70h – 7Fh	7F0h	Accesses 70h – 7Fh
47Fh		4FFh		57Fh		5FFh		67Fh		6FFh		77Fh		7FFh	

Paměť RAM – banky 30 a 31

Bank 30

F0Ch	—
F0Dh	—
F0Eh	—
F0Fh	CLCDATA
F10h	CLC1CON
F11h	CLC1POL
F12h	CLC1SEL0
F13h	CLC1SEL1
F14h	CLC1GLS0
F15h	CLC1GLS1
F16h	CLC1GLS2
F17h	CLC1GLS3
F18h	CLC2CON
F19h	CLC2POL
F1Ah	CLC2SEL0
F1Bh	CLC2SEL1
F1Ch	CLC2GLS0
F1Dh	CLC2GLS1
F1Eh	CLC2GLS2
F1Fh	CLC2GLS3
F20h	CLC3CON
F21h	CLC3POL
F22h	CLC3SEL0
F23h	CLC3SEL1
F24h	CLC3GLS0
F25h	CLC3GLS1
F26h	CLC3GLS2
F27h	CLC3GLS3
F28h	CLC4CON
F29h	CLC4POL
F2Ah	CLC4SEL0
F2Bh	CLC4SEL1
F2Ch	CLC4GLS0
F2Dh	CLC4GLS1
F2Eh	CLC4GLS2
F2Fh	CLC4GLS3
F30h – F6Fh	Unimplemented Read as '0'

Bank 31

F8Ch – FE3h	Unimplemented Read as '0'
FE4h	STATUS_SHAD
FE5h	WREG_SHAD
FE6h	BSR_SHAD
FE7h	PCLATH_SHAD
FE8h	FSR0L_SHAD
FE9h	FSR0H_SHAD
FEAh	FSR1L_SHAD
FEBh	FSR1H_SHAD
FECh	—
FEDh	STKPTR
FEEh	TOSL
FEFh	TOSH

Paměť RAM – Core Registers

INDF – data z nepřímé adresace

- operace s INDF je operací na registru, jehož adresa je ve FSR

PCL, STATUS – viz dále

BSR – Bank Select Register

WREG (nyní už má svou adr.)

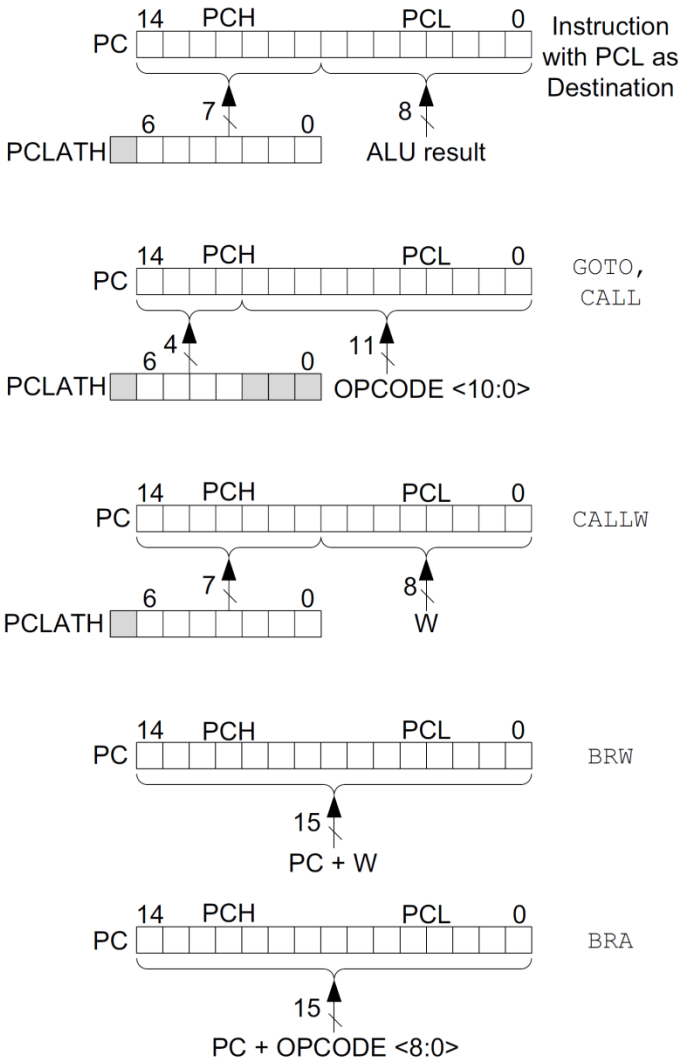
PCLATH – viz dále

INTCON – nastavení přerušení

Addresses	BANKx
x00h or x80h	INDF0
x01h or x81h	INDF1
x02h or x82h	PCL
x03h or x83h	STATUS
x04h or x84h	FSR0L
x05h or x85h	FSR0H
x06h or x86h	FSR1L
x07h or x87h	FSR1H
x08h or x88h	BSR
x09h or x89h	WREG
x0Ah or x8Ah	PCLATH
x0Bh or x8Bh	INTCON

Program Counter (čítač instrukcí)

- PC vždy ukazuje na následující instrukci
- 15-bitový (~ 0 - 7FFFh)
- nižších 8 bitů je pro čtení i zápis přístupno v registru PCL
- horních 7 bitů není přímo přístupno – pro jejich změnu je nutno použít registr PCLATH, jehož obsah se přenáší do vyšších bitů PC při operaci s PCL



STATUS registr (stav, příznaky)

C (Carry/!Borrow) – přenos při sčítání, odečítání a posuvu

DC (Digit carry) – přenos mezi 3. a 4. bitem

Z (Zero) – nulovost aritmetické, logické či přesun. operace

!PD (Power Down) – 0 po instrukci *SLEEP*

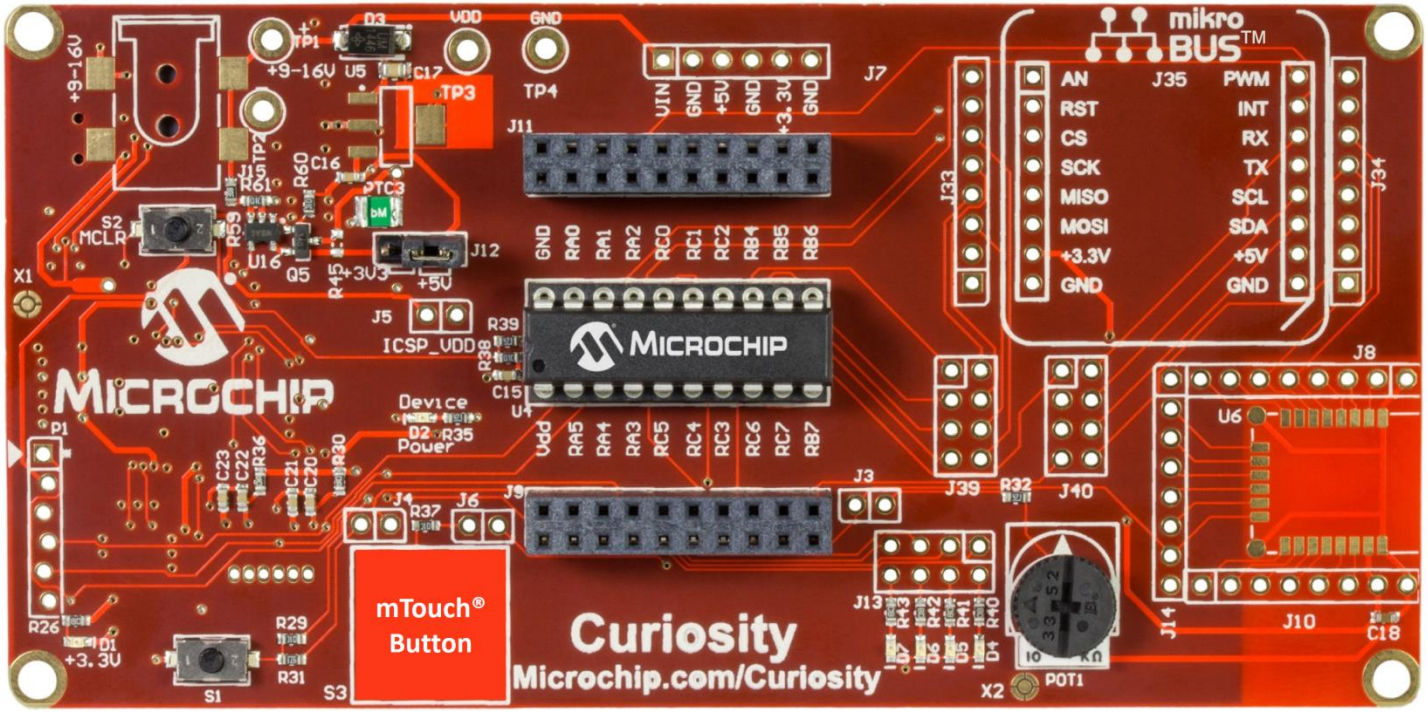
!TO (Time-Out) – 0 po přetečení hlídacího časovače (WDT)

U-0	U-0	U-0	R-1/q	R-1/q	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	$\overline{TO}$	$\overline{PD}$	Z	DC(1)	C(1)
bit 7							bit 0

Microchip Curiosity

Vývojová deska pro vybrané 8b mikrokontrolery PIC

- komunikace s PC přes USB
- integrovaný PICkit3 pro programování a ladění
- pro kompatibilní µC v pouzdrech DIP od 8 do 20 pinů

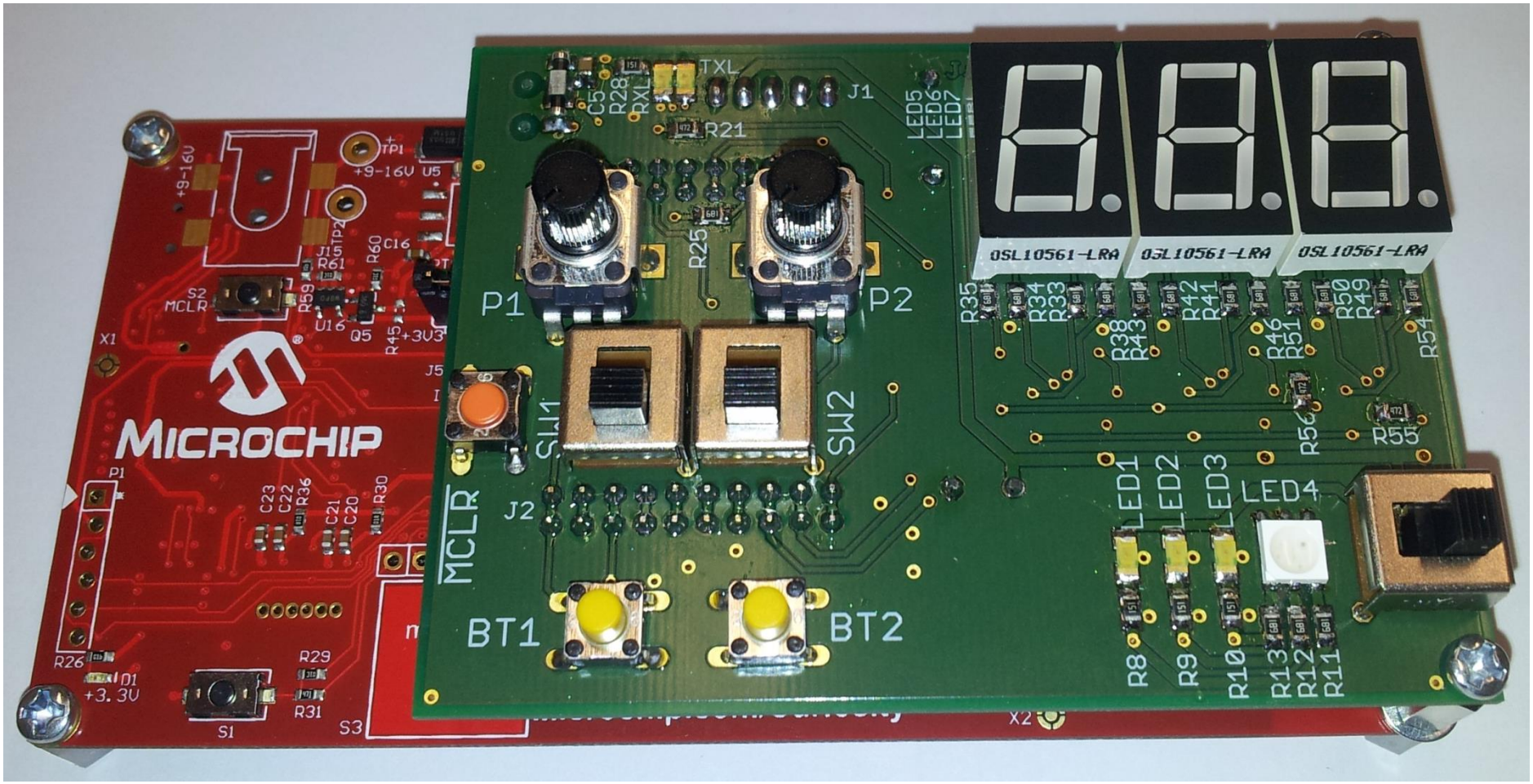


Rozšiřující deska s periferiemi

Výuková deska připojená na Microchip Curiosity

- 2 tlačítka a 2 přepínače
- 3 7-segmentové (+ tečka) LED displeje po SPI
- převodník UART na USB – virtuální COM port
- 3 LED nebo 1 RGB LED (s možností PWM)
- 2 potenciometry pro ADC
- piezo-měnič

Rozšiřující deska s periferiemi



MPLABX IDE

Integrované vývojové prostředí Microchip

- volně ke stažení (bez registrace) pro různé platformy
- pro naše účely stačí IDE + podpora 8b MCU + XC8