Programmable Integrated Controller (PIC)

În cele ce urmează se va prezenta o altă familie de Microcontrolere (similară cu cea prezentată în laboratoarele anterioare), realizată de firma **MICROCHIP TECHNOLOGY INC.** Aceste tipuri de microcontrolere au cunoscut o dezvoltare rapidă într-un scurt interval de timp ca urmare a uşurinţei de programare şi de utilizare a acestora. Programatoarele pentru aceste microcontrolere sunt uşor de realizat şi uşor de interfaţat cu calculatoarele personale. Mai mult decât atât, există controlere care au inclus în interior şi oscilatorul intern, nemaifiind nevoie de rezonator extern. Important de reţinut este că la astfel de controlere bus-ul intern pentru memoria de Date este diferit de cel pentru memoria Program – am avantaj cu lucru simultan pe cele două bus – uri = viteză mai mare de execuţie a instrucţiunilor.

În continuare sunt prezentate câteva trăsături importante ale microcontrolerului **PIC16F84**

- toate instrucţiunile sunt executate într-un singur ciclu, excepţie făcând cele de ramificare
- are doar 35 cuvinte de instrucţiuni
- memorie program accesibilă pe 14 biţi
- memorie de date accesibilă pe 8 biţi
- 15 registre cu funcţii speciale
- stiva pe 8 nivele de adâncime
- moduri de adresare: direct, indirect şi relativ
- 4 surse de întrerupere
- memoria program poate fi reânscrisă de 1.000.000 ori.
- reţinerea datelor în memoria program, fără alimentare, peste 40 ani.
- frecvenţa de lucru maximă 10MHz

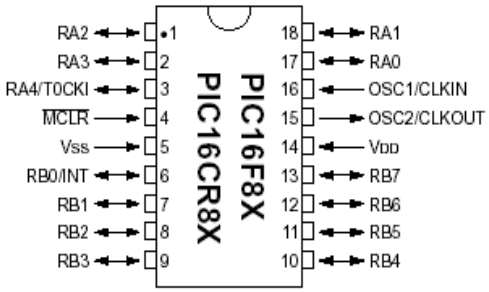


Fig. 1 Vedere de sus a capsulei microcontrolerului PIC16F84

Arhitectura internă pentru PIC16F84 este prezentată în Fig. 2

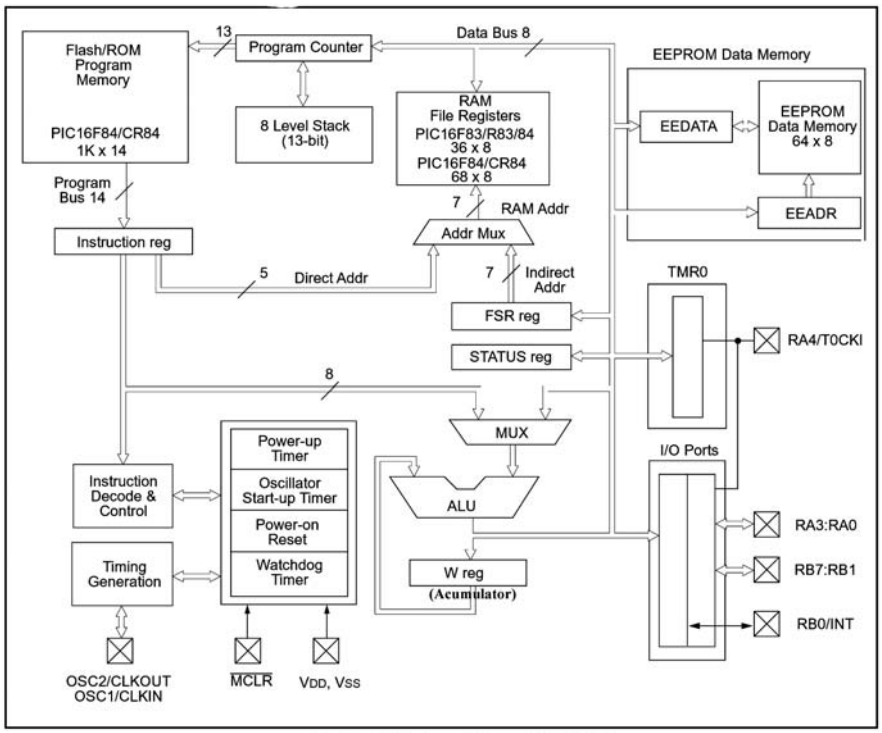


Fig. 2 Arhitectura interna PIC16F84

Memoria Program şi memoria de Date

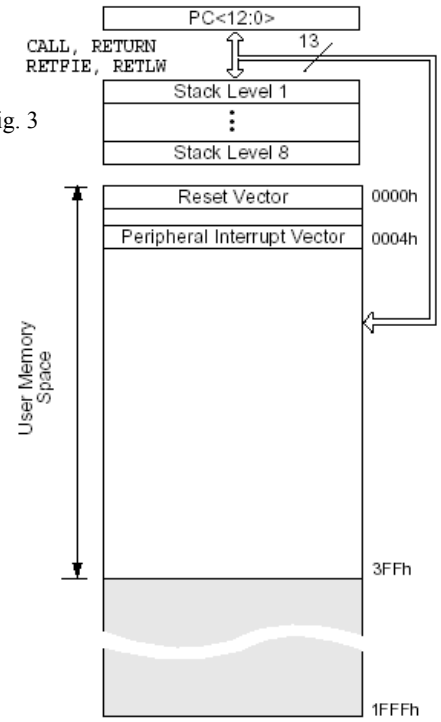


Fig. 3

În Fig. 3 este prezentă harta Memoriei Program. PIC16F84 are PC (Program Counter = numărătorul de program) pe 13 biţi (de la 0 la 12), capabil să acceseze adrese din memoria program până la 8K x 14 biţi. Memoria Program a lui PIC16F84 este de 1K (1024 cuvinte de 14 biţi). (0000H – 03FFH), restul până la 1FFFH este neimplementat.

Litera F din PIC16**F**84 semnifică faptul că Memoria Program poate fi reinscriptibilă. (la PIC12C509, Memoria Program poate fi înscrisă doar *o singură dată*, având avantajul că, odată „arsă" Memoria Program, aşa va rămâne pentru totdeauna – pe când la variantele cu memorie Flash, există varianta teoretică prin care se pot modifica datele (în mod greşit) din Memoria Program.

Stiva

PIC16F84 are o stivă de 13 biţi cu 8 nivele, sau cu alte cuvinte, un grup de 8 locaţii de memorie de 13 biţi lăţime cu funcţii speciale. Rolul său de bază este de a păstra valoarea contorului de program după un salt din programul principal la o adresă a unui subprogram. Pentru ca un program să ştie cum să se întoarcă la punctul de unde a pornit, trebuie să înapoieze valoarea contorului programului din stivă. Când se mută dintr-un program într-un subprogram, contorul programului este împins în stivă (un exemplu de acesta este instrucţiunea CALL). Când se execută instrucţiuni ca RETURN, RETLW sau RETFIE ce au fost executate la sfârşitul unui subprogram, contorul programului a fost luat dintr-o stivă, astfel încât programul să poată continua de unde a fost oprit înainte de a fi întrerupt. Aceste operaţii de plasare într-o şi luare dintr-o stivă de contor de program sunt numite PUSH şi POP.

În Fig. 4 este prezentată toată memoria (programe + date) la PIC16F84.

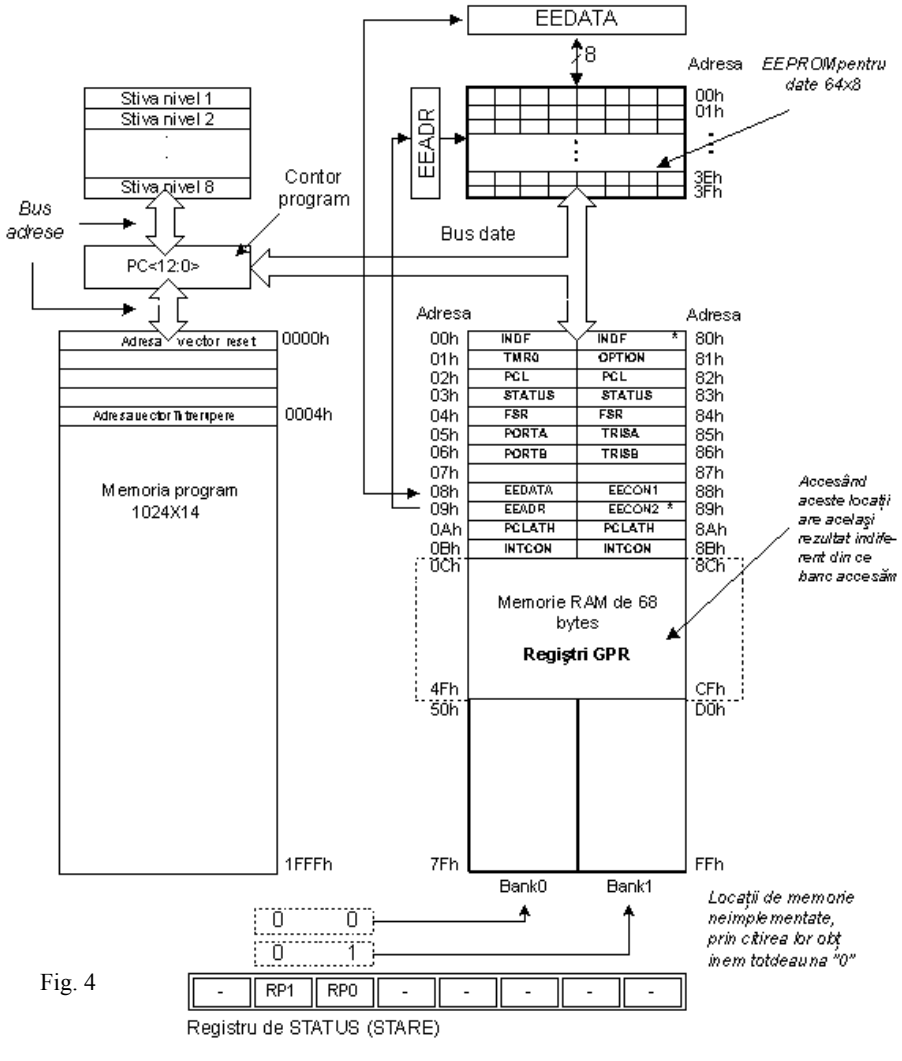


Fig. 4

Organizarea memoriei microcontrolerului PICF84

Memoria de Date este partiţionată în două arii, prima este destinată Regiştrilor cu Funcţii Speciale, iar cea de-a doua fiind de uz general (regiştrii GPR- Global Purpose Register). La rândul ei, zona SFR este şi ea împărţită în două (bank0 şi bank1), selectabile prin valoare lui RP0 (bit aflat în registrul STATUS). Fiecare registru este pe 8 biţi (inclusiv zona EEPROM). Pe lângă SFR şi GPR, mai există şi memoria EEPROM, de 64 de cuvinte pe 8 biţi (64 bytes), accesibilă tot prin regiştrii din SFR. Această zonă de memorie, cu toate că este de Date, nu pierde valoarea, la RESET sau scoaterea de sub tensiune, putându-se stoca aici setări (instantanee) pentru aplicaţia curentă. Zona GPR (General Purpose Register) nu ţine cont de bank0 sau bank1 – valoarea fiind aceeaşi. Zonele 50H – 7Fh şi D0H – FFH nu sunt implementate la microcontrolerul PIC16F84.

În Fig. 5 sunt prezentaţi toţi Regiştrii cu Funcţii Speciale, pentru PIC16F84

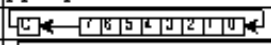
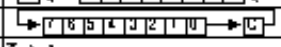
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets (Note3)
Bank 0											
00h	INDF	Uses contents of FSR to address data memory (not a physical register)								---- ----	---- ----
01h	TMR0	8-bit real-time clock/counter								xxxx xxxx	uuuu uuuu
02h	PCL	Low order 8 bits of the Program Counter (PC)								0000 0000	0000 0000
03h	STATUS (2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
04h	FSR	Indirect data memory address pointer 0								xxxx xxxx	uuuu uuuu
05h	PORTA	—	—	—	RA4/T0CKI	RA3	RA2	RA1	RA0	---x xxxx	---u uuuu
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	uuuu uuuu
07h		Unimplemented location, read as '0'								---- ----	---- ----
08h	EEDATA	EEPROM data register								xxxx xxxx	uuuu uuuu
09h	EEADR	EEPROM address register								xxxx xxxx	uuuu uuuu
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of the PC (1)					---0 0000	---0 0000
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
Bank 1											
80h	INDF	Uses contents of FSR to address data memory (not a physical register)								---- ----	---- ----
81h	OPTION	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
82h	PCL	Low order 8 bits of Program Counter (PC)								0000 0000	0000 0000
83h	STATUS (2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
84h	FSR	Indirect data memory address pointer 0								xxxx xxxx	uuuu uuuu
85h	TRISA	—	—	—	PORTA data direction register					---1 1111	---1 1111
86h	TRISB	PORTB data direction register								1111 1111	1111 1111
87h		Unimplemented location, read as '0'								---- ----	---- ----
88h	EECON1	—	—	—	EEIF	WRERR	WREN	WR	RD	---0 x000	---0 q000
89h	EECON2	EEPROM control register 2 (not a physical register)								---- ----	---- ----
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of the PC (1)					---0 0000	---0 0000
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u

Fig. 5

Porturile

Acestei familii de microcontrolere trebuie spus cum să fie Portul (de intrare sau de ieşire). Dacă este de intrare, atunci „citesc" valoarea Portului, iar dacă este de ieşire – scriu în Port. Definirea Porturilor să fie intrare / ieşire se realizează cu ajutorul Regiştrilor – TRISA – pentru PORTUL A şi TRISB – pentru PORTUL B.
Valoare în TRIS = 1 – bitul din PORT este programat ca „intrare"
TRIS = 0 – bitul din PORT este programat ca „ieşire"

În Fig. 6 este prezentat setul de instrucţiuni (35 mnemonice).

Mnemonic		Descriere	Operaţie	Steguleţ	Ciclu	Notă
		Transfer date				
MOVLW	k	Mută constanta în W	k → W		1	
MOVWF	f	Mută W în f	W → f		1	
MOVF	f, d	Mută f	f → d	Z	1	1,2
CLRW	-	Şterge W	00h → W, 1 → Z	Z	1	
CLRF	f	Şterge f	00h → f, 1 → Z	Z	1	2
SWAPF	f, d	Interschimbă nibble-urile în f	f(7:4) → (3:0), f(3:0) → (7:4)		1	1,2
		Aritmetică şi logică				
ADDLW	k	Adună constanta cu W	W+k → W	C,DC,Z	1	
ADDWF	f, d	Adună W cu f	W+f → d	C,DC,Z	1	1,2
SUBLW	k	Scade W din constanta	k-W → W	C,DC,Z	1	
SUBWF	f, d	Scade W din f	f-W → d	C,DC,Z	1	1,2
ANDLW	k	ŞI literal cu W	W.AND.k → W	Z	1	
ANDWF	f, d	ŞI W cu f	W.AND.f → d	Z	1	1,2
IORLW	k	SAU inclusiv constanta cu W	W.OR.k → W	Z	1	
IORWF	f, d	SAU inclusiv W cu f	W.OR.f → d	Z	1	1,2
XORLW	k	SAU exclusiv constanta cu W	W.XOR.k → W	Z	1	1,2
XORWF	f, d	SAU exclusiv W cu f	W.XOR.f → d	Z	1	
INCF	f, d	Incrementează f	f+1 → f	Z	1	1,2
DECF	f, d	Decrementează f	f-1 → f	Z	1	1,2
RLF	f, d	Roteşte la stânga prin Carry		C	1	1,2
RRF	f, d	Roteşte la dreapta prin Carry		C	1	1,2
COMF	f, d	Complement f	$\bar{f}$ → d	Z	1	1,2
		Operaţii cu biţi				
BCF	f, b	Şterge bitul f	0 → f(b)		1	1,2
BSF	f, b	Setează bitul f	1 → f(b)		1	1,2
		Direcţionarea unui debit de program				
BTFSC	f, b	Testează bitul f, Sari dacă este şters	salt dacă f(b)=0		1 (2)	3
BTFSS	f, b	Testează bitul f, Sari dacă este setat	salt dacă f(b)=1		1 (2)	3
DECFSZ	f, d	Decrementează f, Sari dacă este 0	f-1 → d, salt dacă Z=1		1(2)	1,2,3
INCFSZ	f, d	Incrementează f, Sari dacă este 0	f+1 → d, salt dacă Z=1		1(2)	1,2,3
GOTO	k	Du-te la adresă	W.AND.k → W		2	
CALL	k	Apelează subrutina	W.AND.f → d		2	
RETURN	-	Întoarcere din Subrutină	TOS → PC		2	
RETLW	k	Întoarcere cu constanta în W	k → W, TOS → PC		2	
RETFIE	-	Întoarcere din întrerupere	TOS → PC, 1 → GIE		2	
		Alte instrucţiuni				
NOP	-	Fără Operaţii			1	
CLRWDT	-	Şterge Timer-ul Watchdog	0 → WDT, 1 → TO, 1 → PD	$\overline{TO}$, $\overline{PD}$	1	
SLEEP	-	Du-te în mod standby	0 → WDT, 1 → TO, 0 → PD	$\overline{TO}$, $\overline{PD}$	1	

Atenţie:

- d este destinaţie - d = 0 –> rezultatul va fi dus in W
  d = 1 – > rezultatul va fi lăsat în operand (notat cu f)
- b este numărul bitului dintr-un operand (f)
  exp. BTFSS f,4 = testează bitul 5 (LSB - 0,1,2,3,**4**,5,6,7 - MSB) din operandul (f).