helion.pl
KSIĘGARNIA INTERNETOWA · WYDAWNICTWO

IDŹ DO
PRZYKŁADOWY ROZDZIAŁ
SPIS TREŚCI

KATALOG KSIĄŻEK
KATALOG ONLINE
ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK
DODAJ DO KOSZYKA

CENNIK I INFORMACJE
ZAMÓW INFORMACJE O NOWOŚCIACH
ZAMÓW CENNIK

CZYTELNIA
FRAGMENTY KSIĄŻEK ONLINE

Mikroprocesory jednoukładowe PIC

Autor: Stanisław Pietraszek
ISBN: 83-7197-974-6
Format: B5, stron: 412

Książka stanowi kompendium wiedzy na temat popularnych mikroprocesorów PIC z rodziny Base-Line i Mid-Range. Przedstawiono nie tylko opisy samych procesorów, ale również: asembler MPASM, zintegrowane środowisko uruchomieniowe MPLAB, zasady programowania procesorów i przykłady programatorów. Szczególnie duży nacisk położono na opis układów peryferyjnych procesorów w tym interfejsów szeregowych i przetworników A/C. W jednym z rozdziałów przedstawiono typową drogę od pomysłu do realizacji wybranego zadania i kilka mini projektów.

Książka adresowana jest do inżynierów i studentów kierunków: elektronika, automatyka, informatyka i telekomunikacja. Ze względu na powszechną obecność mikroprocesorów w wielu urządzeniach, może okazać się przydatna również dla konstruktorów i projektantów układów elektronicznych.

Książka omawia:

- Schemat blokowy, pamięć, rejestry procesorów
- Układ przerwań i układy sterujące
- Porty, liczniki, pamięć EEPROM
- Interfejsy, przetworniki A/C, układy analogowe
- Listę instrukcji
- Asembler MPASM i zintegrowane środowisko uruchomieniowe MPLAB
- Programowanie procesorów, w tym procedury matematyczne
- Dane techniczne procesorów, parametry i oznaczenia

Autor, Stanisław Pietraszek, jest pracownikiem Instytutu Elektroniki na Wydziale Automatyki, Elektroniki i Informatyki Politechniki Śląskiej w Gliwicach. Prowadzi wykłady z przedmiotów: mikroprocesory jednoukładowe PIC i elektroniczna aparatura medyczna.

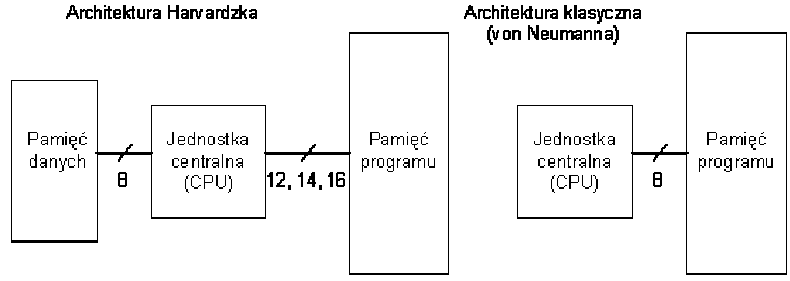
Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Rozdział 1.

Schemat blokowy, pamięć, rejestry

Procesory PIC wykorzystują architekturę Harvardzką, tzn. posiadają rozdzieloną pamięć danych od pamięci programu i dwie oddzielne magistrale o różnej szerokości słowa. Pamięć programu operuje w zależności od rodziny słowem 12-, 14- lub 16-bitowym, natomiast pamięć danych jest 8-bitowa i dlatego procesory PIC zaliczane są do grupy procesorów 8-bitowych. Na rysunku 1.1 pokazano różnice w budowie pomiędzy procesorami wykorzystującymi architekturę tradycyjną, czyli von Neumanna, i Harvardzką.



Rysunek 1.1. *Architektury procesorów*

Dzięki tak długiemu słowu pojedyncza instrukcja może zawierać zarówno kod, jak i argument (argumenty) operacji, co powoduje, że liczba komórek pamięci odpowiada liczbie możliwych do wykonania instrukcji. W przypadku architektury tradycyjnej pierwsza komórka zawiera zwykle kod operacji, a następne argumenty. Powoduje to, że kod wynikowy takich samych programów jest zazwyczaj dla procesorów PIC nieco krótszy.

W procesorach zastosowano potokowy system pracy (*pipeline*), w którym procesor podczas wykonywania kolejnego rozkazu pobiera z pamięci następny, co sprawia wrażenie, że wykonanie rozkazu zajmuje tylko jeden cykl maszynowy.

Wszystkie rejestry specjalne procesora — *SFR* (*Special Function Register*) — odwzorowane są w pamięci RAM i dostępne tak samo jak wszystkie inne komórki pamięci, tzn. poprzez adresowanie bezpośrednie lub pośrednie. Pomysł ten nazwano koncepcją rejestrową (*Register File Concept*).

Inną ciekawą własnością procesorów PIC jest to, że każda komórka pamięci (*rejestr*) może być użyta zarówno jako argument, jak i miejsce, gdzie zostanie umieszczony wynik operacji, tradycyjnie przesyłany do akumulatora. Jego rolę pełni tu rejestr roboczy oznaczony jako *W* (*Working Register*). Brak wyjątków, obecnych zazwyczaj w takich przypadkach, bardzo upraszcza programowanie procesora. Cechę tę nazywa producent *symetrią* lub *ortogonalnością instrukcji*.

Skrócony opis instrukcji dla procesorów z rodziny Mid-Range

Podczas opisu budowy i działania procesorów będziemy często posługiwać się przykładami, stąd już na początku warto, nawet pobieżnie, zaznajomić się z listą rozkazów procesorów PIC. Do tego celu najlepiej nadaje się lista dla procesorów z rodziny *Mid-Range*, zawierająca tylko 35 instrukcji [2]. W przedstawionym w tabeli 1.1 zestawieniu, instrukcje podano w uporządkowaniu funkcjonalnym, odbiegającym od alfabetycznego, spotykanego w dokumentacji procesorów. Pełny opis instrukcji wraz z przykładami znajduje się w rozdziale 5. Tam też znaleźć można listę instrukcji dla procesorów z rodziny *Base-Line*. Dla formalności przypomnijmy, że każda instrukcja dla procesorów z rodziny *Mid-Range* ma postać 14-bitowego słowa i dzieli się na pole określające kod instrukcji i pole zawierające jeden lub dwa argumenty.

Wszystkie instrukcje wykonywane są w jednym cyklu maszynowym, z wyjątkiem instrukcji skoków, które wykonywane są w dwóch cyklach maszynowych.

Tabela 1.1. *Lista rozkazów procesorów z rodziny Mid-Range*

Mnemonik	Arg.	Opis w jęz. ang.	Opis
NOP		No Operation	cykl pusty
CLRW		Clear W	zeruj W
CLRF	f	Clear F	zeruj F
COMF	f, d	Complement F	zaneguj F, wynik prześlij do d
MOVF	f, d	Move F to d	prześlij F do d
MOVWF	f	Move F to W	prześlij W do F
MOVLW	k	Move k to W	prześlij k do W
ADDWF	f, d	Add W to F	dodaj W do F, wynik prześlij do d
ADDLW	k	Add k to W	dodaj k do W
INCF	f, d	Increment F	zwiększ F o 1, wynik prześlij do d
SUBWF	f, d	Subtract W from F	odejmij W od F, wynik prześlij do d
SUBLW	k	Subtract k from W	odejmij k od W
DECF	f, d	Decrement F	zmniejsz F o 1, wynik prześlij do d
ANDWF	f, d	AND W and F	iloczyn logiczny W i F, wynik prześlij do d

Tabela 1.1. *Lista rozkazów procesorów z rodziny Mid-Range – ciąg dalszy*

Mnemonik	Arg.	Opis w jęz. ang.	Opis
ANDLW	k	AND k and W	iloczyn logiczny k i W
IORWF	f, d	OR W with F	suma logiczna W i F, wynik prześlij do d
IORLW	k	OR k with W	suma logiczna k i W
XORWF	f, d	XOR W with F	suma modulo 2 W i F, wynik prześlij do d
XORLW	k	XOR k with W	suma modulo 2 k i W
RLF	f, d	Rotate Left F	przesuń F o 1 bit w lewo, wynik prześlij do d
RRF	f, d	Rotate Right F	przesuń F o 1 bit w prawo, wynik prześlij do d
SWAPF	f, d	Swap nibbles in F	zamień tetrady w F, wynik prześlij do d
BCF	f, b	Bit Clear F	wyzeruj bit b w F
BSF	f, b	Bit Set F	ustaw bit b w F
BTFSC	f, b	Bit Test F, Skip if Clear	jeśli bit b w F = 0, omiń następną instrukcję
BTFSS	f, b	Bit Test F, Skip if Set	jeśli bit b w F = 1, omiń następną instrukcję
INCFSZ	f, d	Inc. F Skip if Zero	zwiększ F o 1, jeśli wynik = 0 omiń następną instr.
DECFSZ	f, d	Dec. F Skip if Zero	zmniejsz F o 1, jeśli wynik = 0 omiń następną instr.
GOTO	k	Go To address k	skok bezwarunkowy do etykiety k
CALL	k	Call subroutine	wywołaj podprogram
RETURN		Return from subroutine	powrót z podprogramu
RETLW	k	Return with k in W	powrót ze stałą k w W
RETFIE		Return From Interrupt	powrót z przerwania
CLRWDT		Clear WDT	zeruj licznik WDT
SLEEP		Go to standby mode	przejdź w stan uśpienia

gdzie:

f — 7-bitowy adres rejestru (0 – 127)

F — zawartość rejestru o adresie f (0 – 255)

W — zawartość rejestru roboczego (0 – 255)

d — adres wyniku operacji (0,1):

gdy d = 0 wynik operacji przesyłany jest do W

gdy d = 1 wynik operacji przesyłany jest do F

b — numer bitu (0 – 7)

k — 8-bitowa stała liczbowa (0 – 255)

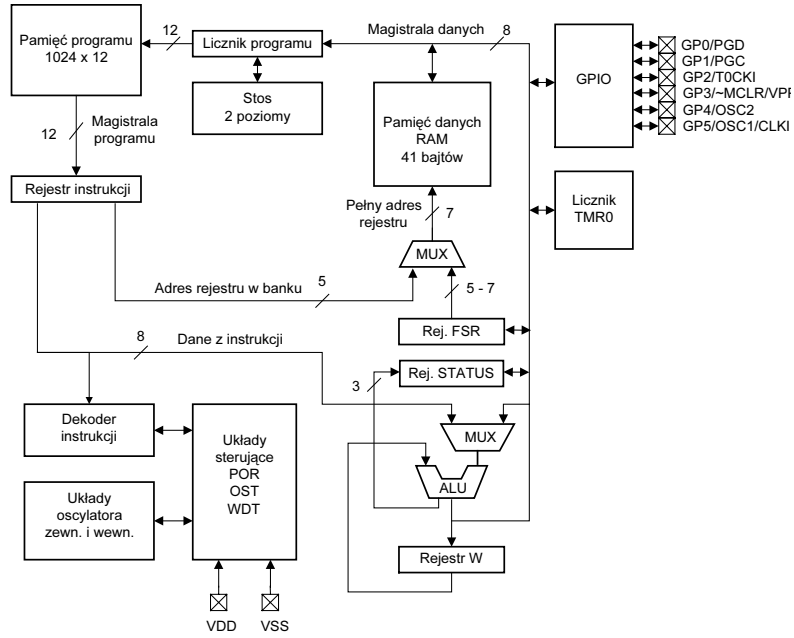
s — 11-bitowa etykieta (0 – 2047)

Schemat blokowy

Na rysunkach 1.2 i 1.3 przedstawiono schematy blokowe procesorów z rodziny *Base-Line* i *Mid-Range* na przykładzie procesorów PIC12C509 [3] i PIC16F877 [4]. Wyróżnić na nich można następujące grupy układów: jednostkę centralną procesora (CPU) wraz z pamięcią programu, pamięcią danych, układami sterującymi i pomocniczymi, uniwersalne porty wejścia/wyjścia oraz układy peryferyjne. W konkretnych typach procesorów wielkość pamięci, liczba portów czy zaimplementowanych układów peryferyjnych jest różna. Podobieństwo struktury pomiędzy rodzinami jest duże, dlatego skupimy się na opisie i analizie układów bardziej złożonych, czyli rodziny *Mid-Range*.

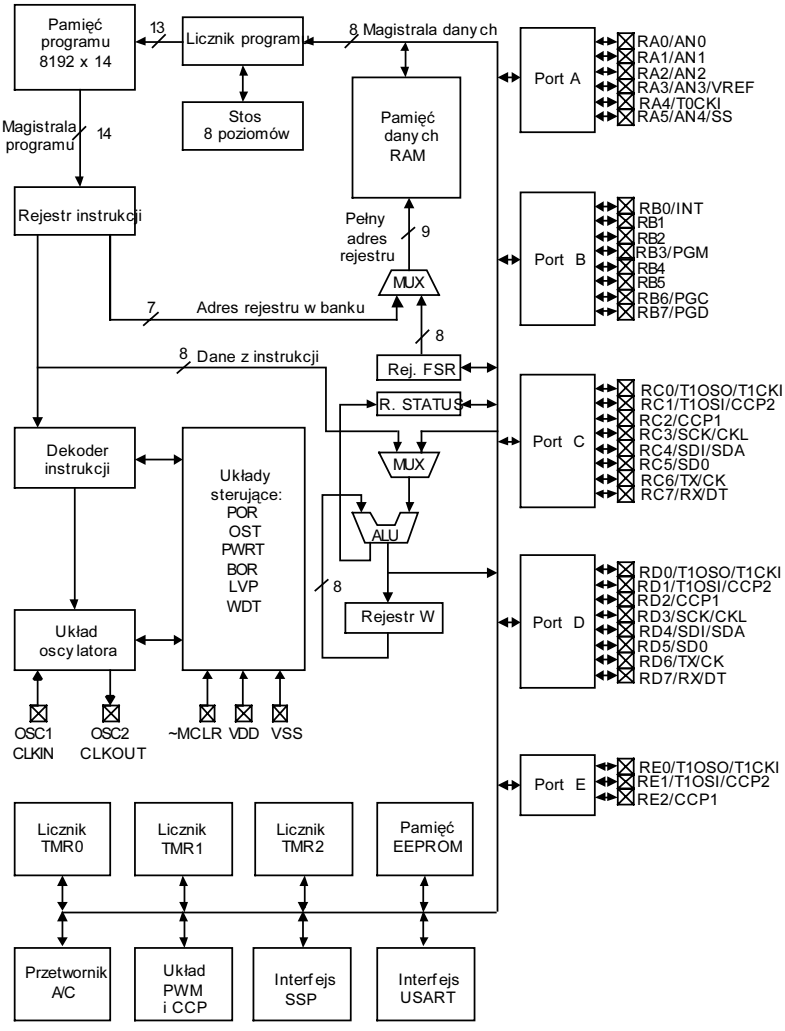
Procesor PIC12C509 należy do grupy najprostszych procesorów, bardzo ubogo wyposażonych w układy peryferyjne i dysponujących względnie małą pamięcią programu — 1024 słów 12-bitowych i małą pamięcią danych (41 rejestrów 8-bitowych w dwóch bankach).

Procesor PIC16F877 należy do grupy procesorów większych z prawie wszystkimi układami peryferyjnymi, pamięcią programu liczącą 8192 słów 14-bitowych i pamięcią danych zawierającą 392 rejestry 8-bitowe w 4 bankach.



Rysunek 1.2. *Schemat blokowy procesora PIC12C509*

Przeglądu procesorów najłatwiej dokonać analizując dane zawarte w broszurze „Product Line Card" [1], wydawanej kilka razy w roku. Zawiera ona zbiorcze zestawienia najważniejszych danych procesorów i umożliwia dobranie procesora najlepiej dopasowanego do aktualnych potrzeb.

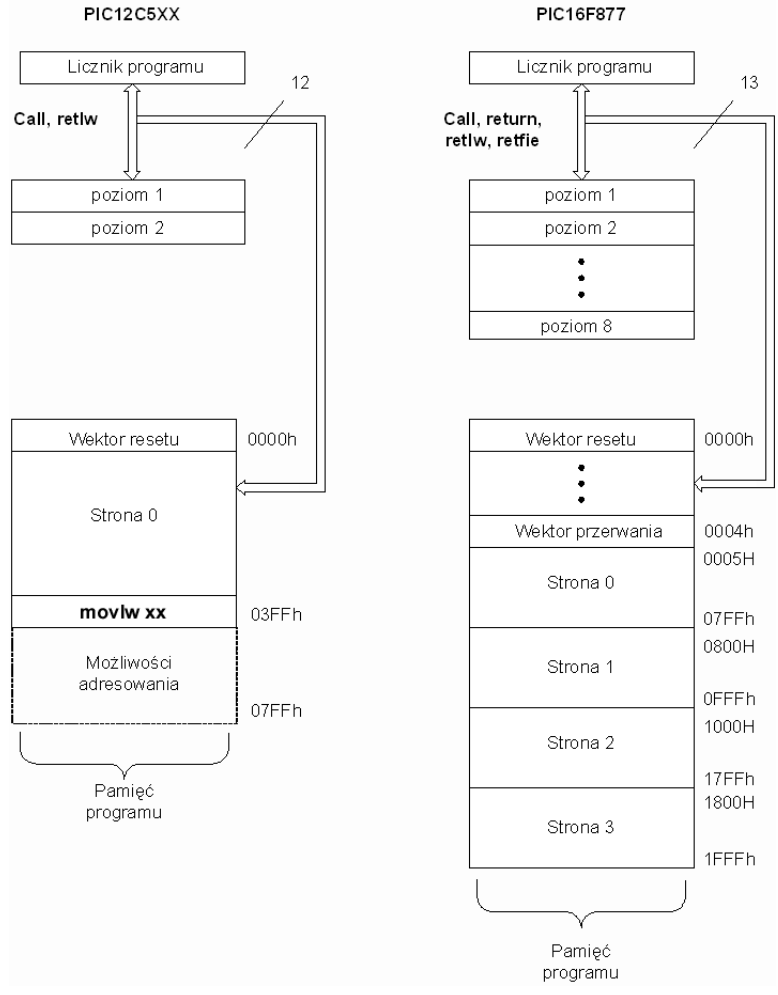


Rysunek 1.3. *Schemat blokowy procesora PIC16F877*

Pamięć programu

Procesory PIC mają wewnętrzną pamięć programu wykonaną w zależności od wersji jako jednokrotnie programowaną pamięć EPROM — wersje C, LC, wielokrotnie programowaną i kasowaną za pomocą promieniowania ultrafioletowego pamięć EPROM — wersja C, końcówka nazwy JW lub programowaną i kasowaną elektrycznie pamięcią FLASH EPROM — wersje LC i LF. Maksymalna wielkość pamięci dla procesorów z rodziny *Mid-Range* wynosi 8 k słów 14-bitowych, o adresach 0000h – 1FFFh. Pamięć adresowana jest za pomocą 13-bitowego licznika rozkazów PC (*Program Counter*) zbudowanego z dwóch 8-bitowych rejestrów PCL — młodszy bajt i PCH — starszy bajt, przy czym trzy najstarsze bity w PCH nie są wykorzystane.

Jeśli mikrokontroler ma fizycznie zaimplementowaną mniejszą pamięć, np. 1 k słów, to adresowanie powyżej tej granicy będzie powodowało wielokrotny dostęp do tych samych słów, o adresach wskazywanych przez 10 młodszych bitów licznika programu. Po wyzerowaniu licznik rozkazów ustawia się na adres 0000h — nazywany wektorem *resetu* — a po pojawieniu się i przyjęciu przerwania na adres 0004h — nazywany wektorem przerwania (rysunek 1.4). W każdym cyklu maszynowym licznik rozkazów jest inkrementowany z wyjątkiem rozkazów skoków, wywołań podprogramów i obsługi przerwań. Sytuacje te opisano bardziej szczegółowo w następnych punktach.

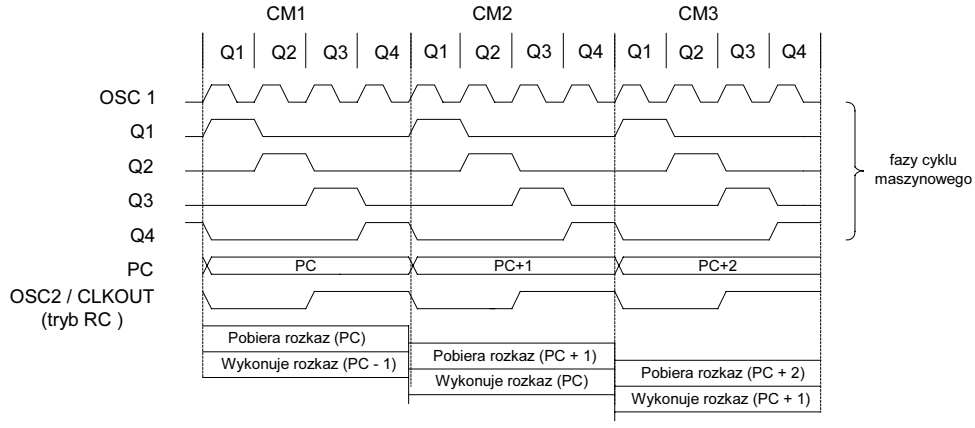


Rysunek 1.4. *Mapy pamięci programu w procesorach Base-Line i Mid-Range*

W procesorach *Base-Line* wielkość pamięci nie może przekraczać 4096 słów 12-bitowych, a licznik programu jest 12-bitowy. Po wyzerowaniu procesor ustawia się w ostatniej komórce pamięci, w której dla procesorów PIC12C5XX znajduje się rozkaz `movlw xx`, gdzie `k` to wartość kalibracyjna dla wewnętrznego oscylatora zegarowego RC. Wykonanie dowolnej, różnej od skoku instrukcji, przenosi nas do komórki 0000h, nazywanej efektywnym adresem resetu. W procesorch *Base-Line* nie ma układu przerwań, stąd adres 004h nie ma specjalnego znaczenia.

Cykl maszynowy

Częstotliwość sygnału zegarowego podawanego na wejście OSC1 jest wewnętrznie dzielona przez cztery, aby wygenerować cztery nie nakładające się na siebie fazy: Q1, Q2, Q3 i Q4, tworzące cykl maszynowy — CM, trwający 4 okresy sygnału wejściowego (rysunek 1.5).



Rysunek 1.5. *Fazy cyklu maszynowego*

W kolejnych fazach następuje:

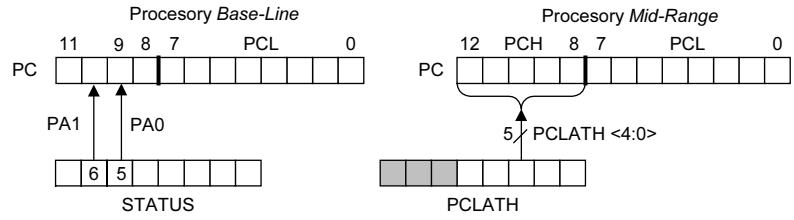
- w fazie Q1 dekodowanie instrukcji, inkrementowanie licznika rozkazów;
- w fazie Q2 czytanie danych;,
- w fazie Q3 wykonanie instrukcji;
- w fazie Q4 zapis danych, pobranie następnej instrukcji.

Jeśli brak jest czynności do wykonania w danej fazie, następuje faza pusta. Licznik rozkazów jest inkrementowany przez każde narastające zbocze sygnału Q1. Instrukcja jest odczytywana z pamięci i zapamiętywana w rejestrze rozkazów podczas trwania fazy Q4, a dekodowana i wykonywana w następnym cyklu, podczas trwania kolejnych faz Q1 do Q4.

Podczas dekodowania i wykonywania jednego rozkazu z pamięci pobierany jest kolejny rozkaz (zapamiętywany w rejestrze rozkazów), stąd wrażenie, że każda instrukcja jest wykonywana w jednym cyklu maszynowym. Jest to słuszne, z wyjątkiem rozkazów skoków, kiedy w instrukcji znajduje się nowy adres komórki pamięci i wstępnie pobrany adres jest nieprzydatny. Wykonanie takich rozkazów zajmuje 2 cykle maszynowe. Sposoby generowania sygnału zegarowego opisano w rozdziale 2.

Licznik programu

W każdym cyklu maszynowym licznik rozkazów jest inkrementowany, z wyjątkiem rozkazów skoków, wywołań podprogramów i obsługi przerwań. Osiem młodszych bitów licznika rozkazów dostępnych jest za pomocą rejestru PCL. Natomiast starsze bity, w zależności od rodziny procesorów i wykonywanego rozkazu, mogą częściowo pochodzić z rejestru STATUS, rejestru PCLATH lub z kodu instrukcji, jak to pokazano na rysunku 1.6.



Rysunek 1.6. *Licznik rozkazów w procesorach Base-Line i Mid-Range*

W procesorach *Base-Line* licznik programu jest 12-bitowy, przy czym dostępnych jest tylko 8 młodszych bitów przez rejestr PCL. Rozkazy skoków bezwzględnych ograniczone są do 512 komórek (poprzez 9-bitowy adres skoku) na aktualnej stronie pamięci, określonej przez bity PA0, PA1 w rejestrze STATUS. Wywołania podprogramów ograniczone są do 256 komórek z powodu 8-bitowego adresu skoku. W procesorze PIC12C509 podczas wywołania podprogramów zerowany jest 8 bit licznika rozkazów, co powoduje, że podprogramy te muszą znajdować się na górnej połowie strony pamięci.

W procesorach *Mid-Range* 13-bitowy licznik programu jest 13-bitowy i zbudowany z dwóch rejestrów PCL i PCH. Młodszy bajt pochodzi z odczytywalnego i zapisywalnego rejestru PCL. Starsze pięć bitów pochodzi z rejestru PCH, który nie jest dostępny w pamięci RAM. Zawartość PCLATH (5 młodszych bitów) jest przenoszona do PCH podczas wykonywania rozkazów skoków. Bardziej szczegółowy opis sposobu aktualizacji zawartości PC w różnych sytuacjach przedstawiono w podrozdziale „Modyfikacja i odtwarzanie zawartości licznika rozkazów".

Stos

Procesory PIC posiadają stos sprzętowy. Nie jest on ani częścią pamięci programu, ani pamięci danych, a wskaźnik stosu nie jest zapisywalny i odczytywalny. Zawartość licznika programu jest zapamiętywana na stosie po wywołaniu instrukcji `call` lub po wystąpieniu przerwania, a odczytywana po wywołaniu instrukcji `return`, `retlw` lub `retfie`, bez żadnych dodatkowych instrukcji. W procesorach *Base-Line* stos ma 2 poziomy, a w procesorach *Mid-Range* — 8 poziomów.

Stos pracuje jako cykliczny bufor, to znaczy, że jeśli zostanie zapisany więcej razy niż wynosi głębokość stosu, to kolejne wartości zostaną zapisane na miejscu pierwszych położonych na stos wartości. Mikrokontroler nie posiada bitów, które informowałyby o przepełnieniu stosu. Operacje odczytu i zapisu na stos nie wpływają na zawartość PCLATH, tzn. nie jest on uaktualniany podczas odtworzenia zawartości PC.

Tryby adresowania, budowa pamięci danych, podział na banki

Dostępne są dwa tryby adresowania: bezpośredni i pośredni. W trybie bezpośrednim adres rejestru w banku pochodzi z instrukcji. W trybie pośrednim odwołujemy się do rejestru INDF. Rejestr INDF nie jest rejestrem fizycznym. Zapisując go, zapisujemy

bajt, którego adres znajduje się aktualnie w rejestrze FSR (rejestr FSR jest w tym przypadku wskaźnikiem do pamięci). Czytanie zawartości INDF bezpośrednio (FSR = 0) zwróci wartość 00h. Zapisywanie INDF bezpośrednio nie da żadnych rezultatów (mimo to zawartość rejestru STATUS może ulec zmianie).

W procesorach PIC pamięć danych zorganizowana jest w bankach (maksimum 4), których wielkość wynika z liczby bitów przeznaczonych na adres w kodzie instrukcji. W procesorach *Base-Line* na adres przeznaczono 5 bitów — stąd banki liczą po 32 rejestry. W procesorach *Mid-Range* adres rejestrów jest 7-bitowy, dlatego banki liczą po 128 rejestrów.

Podział na 4 banki wymaga używania dodatkowych 2 bitów kontrolnych wyboru banku, które znajdują się w rejestrach STATUS i FSR. W tabeli 1.2 przedstawiono kodowanie numeru banku dla obu rodzin procesorów, w dwóch trybach adresowania.

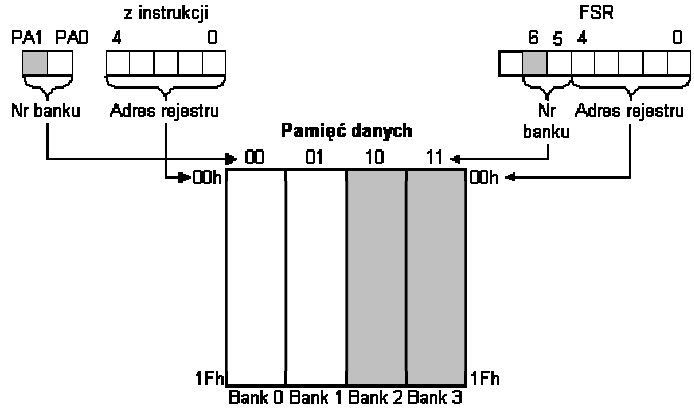
Tabela 1.2. *Adresowanie banków pamięci RAM*

Rodzina	Base-Line				Mid-Range			
Numer banku	Adresowanie bezpośrednie		Adresowanie pośrednie		Adresowanie bezpośrednie		Adresowanie pośrednie	
	PA1	PA0	FSR	<5,6>	RP1	RP0	IRP	FSR<7>
0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0	1
2	1	0	1	0	1	0	1	0
3	1	1	1	1	1	1	1	1

Adresowanie pamięci RAM w procesorach Base-Line

W trybie bezpośrednim adres rejestru pochodzi z instrukcji, a wyboru banku dokonujemy poprzez bity PA1 i PA0 z rejestru STATUS. W niektórych procesorach, np. PIC12C509 wykorzystywane są tylko dwa banki i wtedy bit PA1 pozostaje niezaimplementowany. W procesorze PIC12C508 zastosowano tylko jeden bank (0) i dlatego bit PA0 pozostaje wyzerowany.

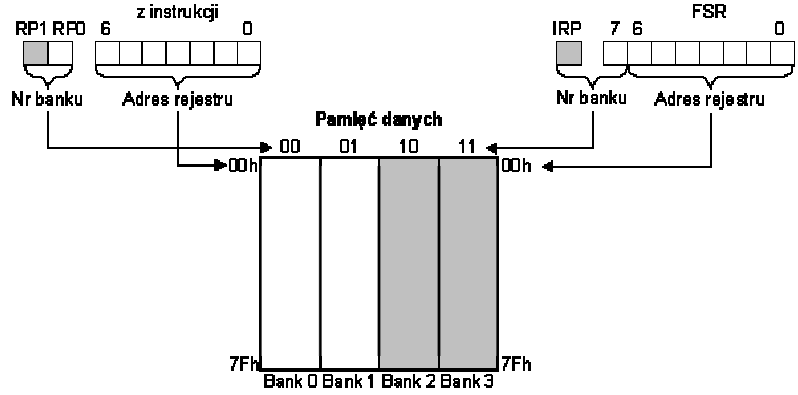
Pośredni tryb adresowania uzyskujemy odwołując się do rejestru INDF i wtedy adres pochodzi z rejestru FSR (6 młodszych bitów). Adres banku określony jest w tym przypadku przez bity 5 i 6, a adres rejestru przez bity 0 – 4 z rejestru FSR. W procesorach PIC12C508 i PIC12C509 nieużywane bity z rejestru FSR przyjmują wartość 1. Na rysunku 1.7 przedstawiono mechanizm tworzenia adresu rejestru dla obydwóch trybów adresowania.



Rysunek 1.7. *Adresowanie bezpośrednie i pośrednie w procesorach Base-Line*

Adresowanie pamięci RAM w procesorach Mid-Range

W trybie bezpośrednim adres rejestru — 7 bitów — pochodzi z instrukcji, a wyboru banku dokonujemy poprzez bity RP1 i RP0 z rejestru STATUS. Pośredni tryb adresowania uzyskujemy odwołując się do rejestru INDF i wtedy adres pochodzi z rejestru FSR — 7 młodszych bitów. Adres banku określony jest w tym przypadku przez 7 bit z rejestru FSR i bit IRP z rejestru STATUS. Na rysunku 1.8 przedstawiono mechanizm tworzenia adresu rejestru dla obydwóch trybów adresowania.



Rysunek 1.8. *Adresowanie bezpośrednie i pośrednie w procesorach Mid-Range*

W niektórych procesorach np. w PIC16F84 wykorzystywane są tylko dwa banki i wtedy bit PA1 oraz bit IRP z rejestru STATUS są niezaimplementowane.

W przykładzie 1.1 w każdej z instrukcji wykorzystano adresowanie bezpośrednie poprzez odwołanie się do rejestrów: STATUS, PORTA i TRISA przez ich nazwy. Ustawianie i zerowanie bitów wyboru banku jest jedną z częściej wykonywanych instrukcji. Warto zwrócić uwagę, że w wyniku wykonania instrukcji `clrf status`, nie zostały

wyzerowane flagi DC, Z, C. W ostatniej linii pokazano ustawienie flagi C w rejestrze STATUS. Podczas wykonywania wielu instrukcji występuje oddziaływanie na zawartość rejestru STATUS, tzn. ustawianie lub zerowanie flag Z, C lub DC.

Przykład 1.1. *Adresowanie bezpośrednie*

```
clrf    status          ;zeruje irp, rp1, rp0
clrf    porta           ;zeruj porta
bsf     status, rp0     ;bank 1
clrf    trisa           ;zeruj trisa
bcf     status, rp0     ;bank 0
bsf     status, c       ;0 -> c, zeruj flage C
```

W przykładzie 1.2 wykorzystano adresowanie pośrednie do wyzerowania zawartości rejestrów o adresach 20h i 21h w banku 0 oraz rejestru TRISB w banku 1.

Przykład 1.2. *Adresowanie pośrednie*

```
clrf    status          ;bank 0
movlw   h'20'           ;20h ->w (adres poczatkowy)
movwf   fsr             ;w -> fsr
clrf    indf            ;zeruj rejestr 20h
incf    fsr, f          ;zwieksz wskaznik
clrf    indf            ;zeruj rejestr 21h
movlw   trisb           ;adres rejestru trisb ->w
movwf   fsr             ;w -> fsr
clrf    indf            ;zeruj rejestr trisb
```

Rejestry specjalne procesora (SFR)

W dolnej części każdego z banków pamięci RAM od adresu 00h znajdują się *rejestry specjalne procesora* (*SFR*). W rodzinie *Mid-Range* zarezerwowano na nie adresy od 00h do 1Fh, ale jest to opcja uwzględniająca wszystkie typy procesorów i często niektóre rejestry są niezaimplementowane. W mniejszych procesorach — np. PIC16F84 — zarezerwowany obszar obejmuje adresy od 00h do 0Ah. W procesorze PIC12C509 zarezerwowany obszar obejmuje adresy 00h – 06h. W tabeli 1.3 przedstawiono mapę pamięci procesora PIC16F877, a w tabelach 1.4, 1.5 i 1.6 mapy pamięci następujących procesorów: PIC16F628, PIC16F84 i PIC12C509.

Niektóre ważniejsze rejestry procesora są odwzorowane we wszystkich bankach, tzn. dostęp do nich jest możliwy niezależnie od aktualnie wybranego banku. Warto zwrócić uwagę, że ze względu na kompatybilność programową pomiędzy procesorami starano się utrzymać zasadę, że niezależnie od ilości zaimplementowanych rejestrów te same rejestry mają stałe adresy. Mapie pamięci warto poświęcić więcej uwagi, ponieważ podczas pisania programów jest ona bardzo często wykorzystywana.

Tabela 1.3. *Mapa pamięci RAM w procesorze PIC16F877*

Bank 0	Adres rejestru	Bank 1	Adres rejestru	Bank 2	Adres rejestru	Bank 3	Adres rejestru
INDF	00h	**INDF**	80h	**INDF**	100h	**INDF**	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	**PCL**	82h	**PCL**	102h	**PCL**	182h
STATUS	03h	**STATUS**	83h	**STATUS**	103h	**STATUS**	183h
FSR	04h	**FSR**	84h	**FSR**	104h	**FSR**	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD	08h	TRISD	88h		108h		188h
PORTE	09h	TRISE	89h		109h		189h
PCLATH	0Ah	**PCLATH**	8Ah	**PCLATH**	10Ah	**PCLATH**	18Ah
INTCON	0Bh	**INTCON**	8Bh	**INCON**	10Bh	**INCON**	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	*	18Eh
TNR1H	0Fh		8Fh	EEADRH	10Fh	*	18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h	GPR	117h	GPR	197h
RCSTA	18h	TXSTA	98h	16 rej.	118h	16 rej.	198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch		9Ch		11Ch		19Ch
CCP2CON	1Dh		9Dh		11Dh		19Dh
ADRESH	1Eh	ADRESL	9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
Ogólnie dostępne (GPR) 96 rej.	20h 7Fh	GPR 80 rej. Zamap. jako 70h – 7Fh 16 rej.	A0h EFh F0h FFh	GPR 80 rej. Zamap. jako 70h – 7Fh 16 rej.	120h 16Fh 170h 17Fh	GPR 80 rej. Zamap. jako 70h – 7Fh 16 rej.	1A0h 1EFh 1F0h 1FFh

Wyróżnione rejestry dostępne są niezależnie od banku

* Rejestry zarezerwowane dla ICD (*In Circuit Debugger*)

Tabela 1.4. *Mapa pamięci RAM w procesorze PIC16F628*

Bank 0	Adres rejestru	Bank 1	Adres rejestru	Bank 2	Adres rejestru	Bank 3	Adres rejestru
INDF	00h	**INDF**	80h	**INDF**	100h	**INDF**	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	**PCL**	82h	**PCL**	102h	**PCL**	182h
STATUS	03h	**STATUS**	83h	**STATUS**	103h	**STATUS**	183h
FSR	04h	**FSR**	84h	**FSR**	104h	**FSR**	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
	07h		87h		107h		187h
	08h		88h		108h		188h
	09h		89h		109h		189h
PCLATH	0Ah	**PCLATH**	8Ah	**PCLATH**	10Ah	**PCLATH**	18Ah
INTCON	0Bh	**INTCON**	8Bh	**INCON**	10Bh	**INCON**	18Bh
PIR1	0Ch	PIE1	8Ch		10Ch		18Ch
PIR2	0Dh		8Dh		10Dh		18Dh
TMR1L	0Eh	PCON	8Eh		10Eh		18Eh
TNR1H	0Fh		8Fh		10Fh		18Fh
T1CON	10h		90h		110h		190h
TMR2	11h		91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
	13h		93h		113h		193h
	14h		94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h		117h		197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah	EEDATA	9Ah		11Ah		19Ah
	1Bh	EEADR	9Bh		11Bh		19Bh
	1Ch	EECON1	9Ch		11Ch		19Ch
	1Dh	EECON2	9Dh		11Dh		19Dh
	1Eh		9Eh		11Eh		19Eh
CMCON	1Fh	VRCON1	9Fh		11Fh		19Fh
Ogólnie dostępne 96 (GPR)	20h 7Fh	80 GPR Zamap. jako 70h – 7Fh	A0h EFh F0h FFh	48 GPR Zamap. jako 70h – 7Fh	120h 14Fh 170h 17fH	 Zamap. jako 70 – 7Fh	1A0h 1EFh 1F0h 1FFh

Tabela 1.5. *Mapa pamięci RAM w procesorze PIC16F84*

Bank 0	Adres rejestru	Bank 1	Adres rejestru
INDF	00h	INDF	80h
TMR0	01h	OPTION_REG	81h
PCL	02h	PCL	82h
STATUS	03h	STATUS	83h
FSR	04h	FSR	84h
PORTA	05h	TRISA	85h
PORTB	06h	TRISB	86h
	07h		87h
EEDATA	08h	EECON1	88h
EEADR	09h	EECON2	89h
PCLATH	0Ah	PCLATH	8Ah
INTCON	0Bh	INTCON	8Bh
Ogólnie dostępne (GPR)	0CH	Zamap. w banku 0	8Ch
	0Dh		8Dh
	0Eh		8Eh
	0Fh		8Fh
68 rej.	4Fh		CFh
	50h		D0h
	7Fh		FFh

Mapa pamięci procesora PIC16F877 jest charakterystyczna dla procesorów z większą liczbą układów peryferyjnych. Chociaż większość rejestrów specjalnych w różnych procesorach ma te same adresy, warto zwrócić uwagę na parę różnic. W procesorach z przetwornikiem A/C 8-bitowym rejestr ADRESH nazywa się ADRES. W procesorach z układem komparatorów na miejscu rejestrów ADCON0 i ADCON1 znajdują się rejestry CMCON i VRCON. W procesorach PIC16F873 rejestry GPR o adresach 120h – 17Fh mapowane są w banku 0, a rejestry o adresach 1A0h – 1FFh mapowane są w banku 1.

Rejestry ogólnego przeznaczenia (GPR)

Pozostałą część pamięci RAM zajmują *rejestry ogólnego przeznaczenia* (*GPR*). Liczba ich zależy od typu procesora i wynosi od 25 do 368. W niektórych procesorach — np. PIC16F87X — ogólnie dostępne rejestry o adresach względnych 70h – 7Fh w bankach 1, 2, 3 zamapowane są w banku 0, tzn. odwołanie się do nich powoduje, że operacja wykonywana jest na rejestrze o tym samym adresie względnym w banku 0. Niezaimplementowane rejestry z banku 1 są niekiedy mapowane w banku 0.

Tabela 1.6. *Mapa pamięci RAM w procesorze PIC12C509*

Bank 0	Adres rejestru	Bank 1	Adres rejestru
INDF	00h	Zamap. w banku 0	20h
TMR0	01h		21h
PCL	02h		22h
STATUS	03h		23h
FSR	04h		24h
OSCCAL	05h		25h
GPIO	06h		26h
Ogólnie dostępne (GPR) 9 rej.	07h 0Fh		27h 2Fh
Ogólnie dostępne (GPR) 16 rej	10h 1Fh	Ogólnie dostępne (GPR) 16 rej.	30h 3Fh

Rejestr STATUS

W procesorach *Mid-Range* rejestr STATUS (*statusowy*) zawiera bity określające stan jednostki arytmetyczno-logicznej (ALU), czyli tzw. flagi Z, DC i C, bity wyboru banku IRP, RP0, RP1 oraz bity: ~TO i ~PD, które zależą od okoliczności resetu. Bity Z, DC i C charakteryzują wynik operacji arytmetycznej lub logicznej. Jest on — tak jak każdy rejestr — zapisywalny, tzn. można do niego przesłać wynik wykonania instrukcji. W takim przypadku modyfikacja bitów Z, DC i C jest zablokowana.

W procesorze PIC12C509 w rejestrze STATUS bit 6 jest niezaimplementowany, inne jest też znaczenie bitu 7, który pełni funkcję flagi ustawianej przy zmianie stanu na liniach GP2:0.

Rejestr STATUS (adres 03h, zaimplementowany we wszystkich bankach)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	**RP1**	**RP0**	**~TO**	**~PD**	**Z**	**DC**	**C**
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0

gdzie:

R — bit odczytywalny

W — bit zapisywalny

U — bit niezaimplementowany

n — stan po włączeniu zasilania (0, 1, lub x — stan nieokreślony)

Oznaczenia te stosowane będą przy opisie wszystkich rejestrów.

bit 7 (**IRP**) — bit wyboru banku dla adresowania pośredniego

0 — Bank 0, 1 (0h – FFh)

1 — Bank 2, 3 (100h – 1FFh)

Dla procesorów używających tylko banku 0 i 1 bit 7 IRP nie jest używany i powinien pozostać wyzerowany.

bity 6 – 5 (**RPl:RP0**) — bity wyboru banku dla adresowania bezpośredniego

00 — Bank 0 (00h – 7Fh)

01 — Bank 1 (80h – FFh)

10 — Bank 2 (100h –17Fh)

11 — Bank 3 (180h –1FFh)

Każdy bank ma 128 bajtów. Dla procesorów używających tylko banku 0 i 1 bit 6 (**RP1**) nie jest używany i powinien pozostać wyzerowany.

bit 4 (**~TO**) — bit przepełnienia licznika WDT

1 — po włączeniu, wykonaniu instrukcji `clrwdt` lub `sleep`

0 — nastąpiło przepełnienie licznika WDT

bit 3 (**~PD**) — bit sygnalizujący przejście w stan uśpienia

1 — po włączeniu lub wykonaniu instrukcji `clrwdt`

0 — po wykonaniu instrukcji `sleep`

bit 2 (**Z**) — bit (flaga) zera

1 — wynikiem operacji arytmetycznej lub logicznej jest zero

0 — wynikiem operacji arytmetycznej lub logicznej jest liczba różna od zera

bit 1 (**DC**) — bit przeniesienia połówkowego (*Digit Carry*)

1 — nastąpiło przeniesienie z 3 na 4 bit

0 — nie nastąpiło przeniesienie z 3 na 4 bit

bit 0 (**C**) — bit przeniesienia/pożyczki (*Carry*)

1 — nastąpiło przeniesienie z najbardziej znaczącego bitu

0 — nie nastąpiło przeniesienie z najbardziej znaczącego bitu

Dla operacji odejmowania (instrukcje: `subwf` i `sublw`) bit C pełni funkcję zanegowanej pożyczki.

Rejestr STATUS dla procesora PIC12C509 (adres 03h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
GPWUF	**–**	**PA0**	**~TO**	**~PD**	**Z**	**DC**	**C**
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0

bit 7 (**GPWUF**) — bit (flaga) ustawiana przy zmianie stanu na liniach GP3, 1, 0

0 — po wyzerowaniu procesora po załączeniu zasilania (POR), z WDT i ~MCLR

1 — po wyzerowaniu procesora po zmianie na liniach GP3, 1, 0 (od ostatniego odczytu)

bit 5 (**PA0**) — bit wyboru banku

0 — bank 0

1 — bank 1

W przykładzie 1.3 pokazano kilka typowych rozkazów wykonywanych na rejestrze STATUS. Zastosowano instrukcję `clrf status` do wyzerowania zawartości rejestru STATUS. Jak już wcześniej wspomniano, nie odnosi się ona do bitów Z, DC i C. Kolejne linie pokazują, jak można ustawić bity RP0 i RP1 oraz flagę przeniesienia (C). Ostatni fragment pokazuje sprawdzanie, czy zawartość rejestru `reg1` równa jest zero, przez sprawdzenie flagi Z, ustawianej podczas wykonywania instrukcji `movf reg1, f`.

Przykład 1.3. *Operacje na rejestrze STATUS*

```
clrf    status          ;zeruj zawartosc rejestru status, z,dc,c bez zmian
bsf     status, rp0     ;1 -> rp0
bsf     status, rp1     ;1 -> rp1, bank 3
bsf     status, c       ;1 -> c, (ustaw flage c)

clrf    status          ;bank 0
movf    reg1, f         ;odczytaj reg1
btfss   status, z       ;sprawdz flage z
goto    xxx             ;skok do xxx gdy reg1=0,(z=0)
goto    yyy             ;skok do yyy gdy reg1=0,(z=1)
```

Modyfikacja i odtwarzanie zawartości licznika rozkazów

Relacje pomiędzy licznikiem rozkazów, stosem a rejestrem STATUS najlepiej prześledzić na przykładzie opisu zachowania się procesora w typowych sytuacjach obejmujących rozkazy odwołujące się do rejestru PCL, rozkazy skoków i wywołań podprogramów. Przedstawione opisy posłużą w tym miejscu jedynie do ilustracji mechanizmów modyfikacji i odtwarzania licznika rozkazów. Przykłady wykorzystujące te mechanizmy podano przy okazji opisu rozkazów skoków w rozdziale 5.

Modyfikacja i odtwarzanie zawartości PC w procesorach Mid-Range

Skok wyliczany (computed goto)

Modyfikacja zawartości rejestru PCL równoważna jest wykonaniu skoku i nazywana jest skokiem wyliczanym. Skok taki odbywa się wewnątrz ciągłego obszaru pamięci obejmującego 256 komórek o adresach wskazywanych przez zawartość rejestrów PCL i PCLATH. Często modyfikacja PCL polega na zmianie go o pewną wartość i wtedy może być wymagane sprawdzenie, czy adresy skoków nie przekraczają granic 256-bajtowych obszarów, lub też posługiwanie się pełnym adresem, tj. modyfikacją PCL i PCLATH (sytuacja 1). Rozkazy te nie wpływają na zawartość stosu.

Skok bezwzględny

Podczas wykonania instrukcji `goto` do rejestru PCL i dwóch najmłodszych bitów PCH wpisywanych jest 11 bitów z kodu instrukcji, a dwa najstarsze bity w PCH przepisywane są z 3 i 4 bitu PCLATH. Fakt, że tylko 11 młodszych bitów pochodzi z kodu instrukcji, ogranicza zakres skoku do bloku 2k pamięci, w której się aktualnie znajdujemy (sytuacja 2. na rysunku 1.9).

Wywołanie podprogramu

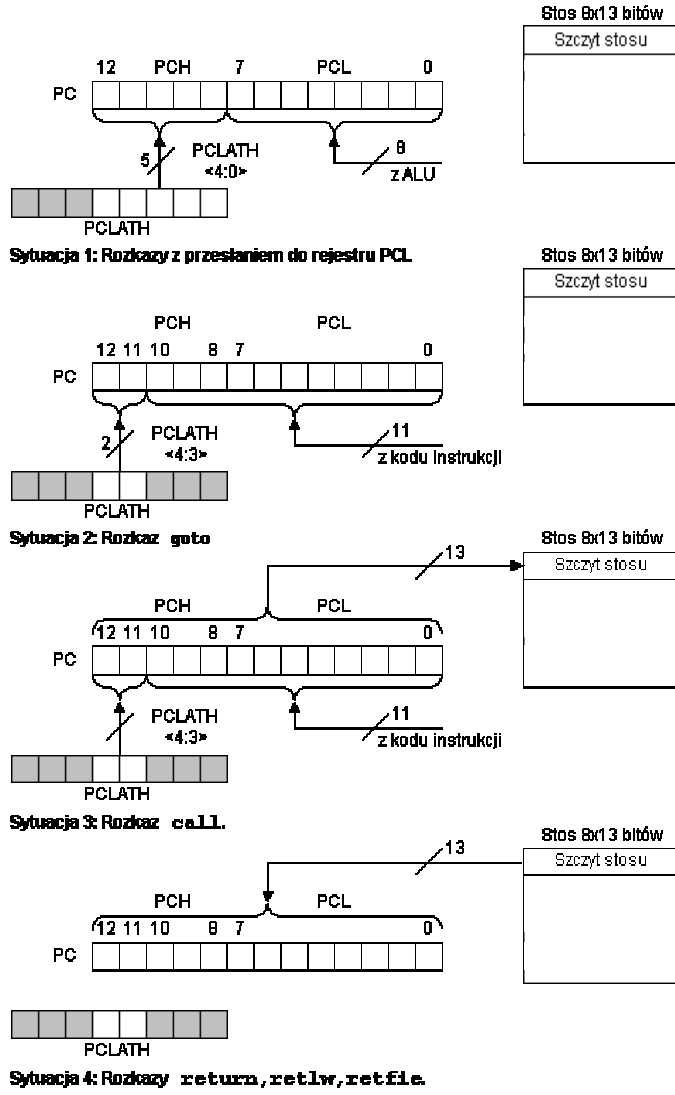
Podczas wywołania podprogramu (`call`) aktualna wartość licznika rozkazów (13 bitów) odkładana jest na stos, a do licznika rozkazów zostaje wpisanych 11 bitów z kodu instrukcji (adres skoku) i 2 bity z rejestru PCLATH (bit 3 i 4). Ogranicza to zakres skoku do bloku 2k pamięci, w którym się aktualnie znajdujemy — sytuacja 3. na rysunku 1.9.

Powrót z podprogramu i z procedury obsługi przerwania

Przy powrocie z podprogramu lub z procedury obsługi przerwania (instrukcje: `retutn`, `retlw`, `retfie`) zawartość szczytu stosu (13 bitów) jest przesyłana do licznika rozkazów, nie jest jednak aktualizowana zawartość rejestru PCLATH — (sytuacja 4. na rysunku 1.9). Przykład 1.4 ilustruje mechanizm przechodzenia pomiędzy stronami pamięci na przykładzie wywołania ze strony 0 procedury `del` znajdującej się na stronie 1 pamięci.

Przykład 1.4. *Zmiana adresu strony*

```
org     h'200'          ;strona 0
bcf     pclath, 4       ;przygotowanie adresu strony 1
bsf     pclath, 3       ;pclath<4,3>=01
call    del             ;wywolanie procedury na stronie 1
;...
org     h'900'          ;strona 1
del                     ;procedura del na stronie 1
;...
return
```

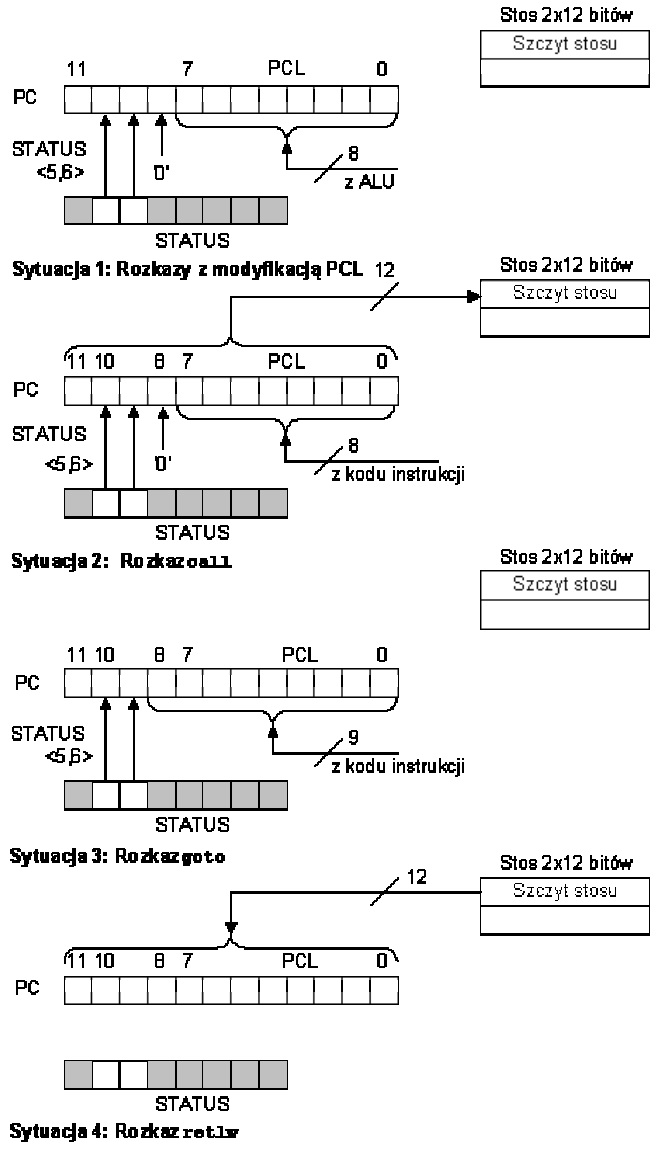


Rysunek 1.9. *Modyfikacja i odtwarzanie licznika rozkazów w procesorach Mid-Range*

Modyfikacja i odtwarzanie zawartości PC w procesorach Base Line

W procesorach *Base-Line* bity wpisywane do licznika rozkazów pochodzą z kodu instrukcji i z rejestru STATUS (bity PA1 i PA0). Bardziej szczegółowy opis tej sytuacji dla procesora PIC12C509 przedstawiono na rysunku 1.10.

Warto zwrócić uwagę, że podczas wykonywania instrukcji odwołującej się do rejestru PCL zerowany jest 8 bit licznika rozkazów — sytuacja 1. na rysunku 1.10, co powoduje ograniczenie zakresu skoku do młodszych 256 bajtów na aktualnej stronie pamięci. Podobna sytuacja występuje przy wywołaniu podprogramu — sytuacja 2.



Rysunek 1.10. *Modyfikacja i odtwarzanie licznika rozkazów w procesorach Base-Line*

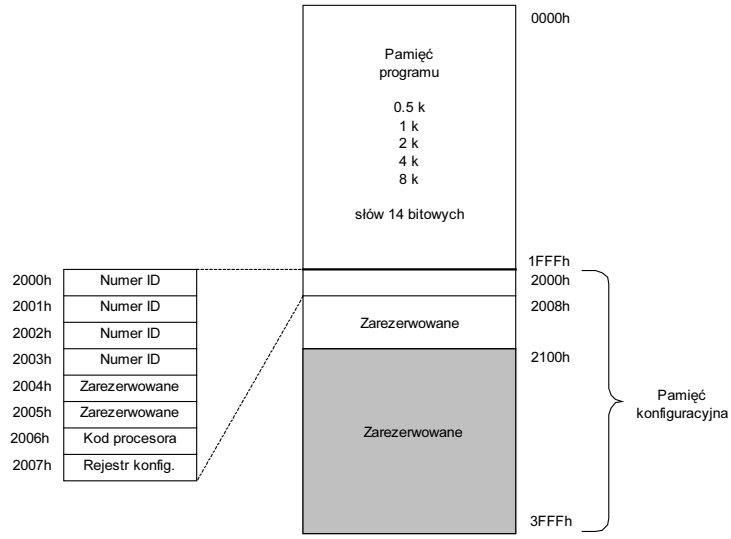
Pamięć konfiguracyjna

W procesorach PIC poza obszarem pamięci programu znajduje się *pamięć konfiguracyjna*. Nominalnie ma ona taki sam rozmiar jak pamięć programu, ale zaimplementowanych jest tylko część komórek. Dostęp do pamięci konfiguracyjnej jest możliwy tylko w czasie programowania, a wyjście z niej następuje tylko poprzez wyzerowanie procesora. Warto zwrócić uwagę, że adres pamięci konfiguracyjnej znajduje się poza zakresem adresowania przez licznik programu.

Na uwagę zasługują komórki o adresach 2000h do 2007h. Komórka o adresie 2007h pełni tu szczególną rolę i nazywana jest rejestrem konfiguracyjnym procesora. Ważną rolę pełni też adres 2100h, który jest logicznym adresem początku pamięci EEPROM.

Na rysunku 1.11 przedstawiono mapę pamięci konfiguracyjnej w procesorach z rodziny *Mid-Range*. Zawartość rejestru konfiguracyjnego i znaczenie poszczególnych bitów zmienia się w zależności od rodziny procesorów. Bardziej szczegółowy opis rejestru konfiguracyjnego przedstawiony jest w dalszej części tego rozdziału, a sposób programowania w rozdziale 8.

Rysunek 1.11. *Mapa pamięci konfiguracyjnej w procesorach Mid-Range*



W procesorach PIC508 i PIC509 rejestry z numerem ID znajdują się w pierwszych komórkach 64-bitowego fragmentu pamięci konfiguracyjnej, a rejestr konfiguracyjny znajduje się w ostatniej komórce tego fragmentu o adresie FFF. Mapę pamięci konfiguracyjnej dla tych procesorów pokazano na rysunku 1.12.

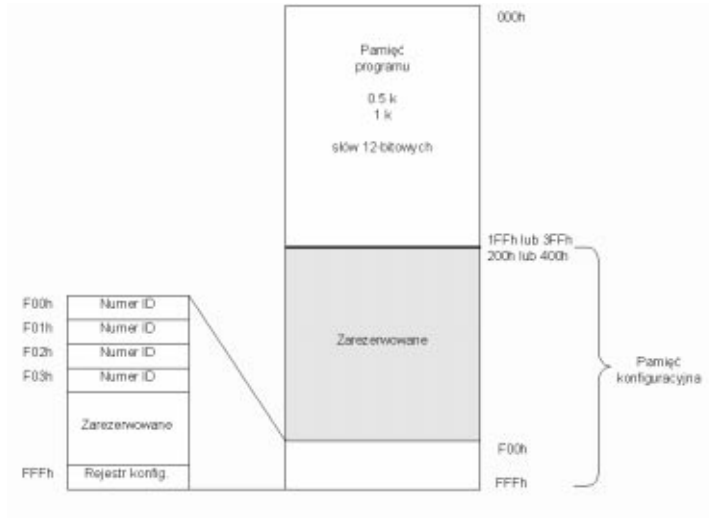
Zawartość pamięci konfiguracyjnej

Zawartość pierwszych ośmiu komórek pamięci konfiguracyjnej w procesorach *Mid-Range* przedstawiono na rysunku 1.13. Bity dostępne dla użytkownika zaznaczono kolorem szarym.

Numer ID

Zawartość słów 2000h – 2003h przeznaczona jest do zapisu *numeru identyfikacyjnego procesora* (ID), przy czym wykorzystywać można tylko 4 najmłodsze bity z każdego ze słów. Pozwala to na zapisanie 16 bitów numeru identyfikacyjnego. Bity te są niedostępne podczas normalnej pracy procesora, ale można je zapisać i odczytać podczas programowania i weryfikacji zawartości pamięci programu.

Słowa o adresach 2004h i 2005h są zarezerwowane.



Rysunek 1.12. *Mapa pamięci konfiguracyjnej w procesorach PIC508 i PIC509*

13	12	11	10	9	8	7	6	5	4	3	2	1	0	adres
Numer identyfikacyjny, tylko 4 najmłodsze bity w każdym słowie: razem 16 bitów														2000h 2001h 2002h 2003h
Zarezerwowane														2004h 2005h
Kod typu procesora														2006h
Rejestr konfiguracyjny														2007h

Rysunek 1.13. *Pamięć konfiguracyjna w procesorach Mid-Range*

Kod typu procesora

W komórce o adresie 2006h znajduje się kod typu procesora.

Rejestr konfiguracyjny

W komórce o adresie 2007h znajdują się *bity konfiguracyjne*. Przed zaprogramowaniem i po skasowaniu całej pamięci programu, we wszystkich komórkach pamięci — w tym i pamięci konfiguracyjnej — znajduje się wartość 1. W czasie programowania następuje przeprogramowanie wybranych bitów na 0.

Zawartość komórki 2007h, nazywanej *rejestrem konfiguracyjnym*, może być różna dla różnych procesorów. Dla przykładu podano zawartość tego rejestru dla kilku procesorów i opisano znaczenie poszczególnych bitów.

Rejestr konfiguracyjny w procesorze PIC16F877

13	12	11	10	9	8	7	6	5	4	3	2	1	0
CP1	CP0	DEBUG	–	WRT	CPD	LVP	BODEN	CP1	CP0	~PWRTE	WDTE	FOSC1	FOSC0

Rejestr konfiguracyjny w procesorze PIC16F628

13	12	11	10	9	8	7	6	5	4	3	2	1	0
CP1	CP0	CP1	CP0	–	CPD	LVP	BODEN	MCLRE	FOSC2	~PWRTE	WDTE	FOSC1	FOSC0

Rejestr konfiguracyjny w procesorze PIC16F84

13	12	11	10	9	8	7	6	5	4	3	2	1	0
CP	CP	CP	CP	CP	CP	CPD	CP	CP	CP	~PWRTE	WDTE	FOSC1	FOSC0

Rejestr konfiguracyjny w procesorze PIC12C509

11	10	9	8	7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	MCLRE	CP	WDTE	FOSC1	FOSC0

bity **FOSC1**, **FOSC0** — bity wyboru typu oscylatora (dla procesorów bez bitu **FOSC2**)

11 — oscylator RC

10 — oscylator HS

01 — oscylator XT

00 — oscylator LP

bity **FOSC2**, **FOSC1**, **FOSC0** — bity wyboru typu oscylatora (dla procesorów z bitem **FOSC2**)

111 — ER lub RC z wyjściem OSCOUT

110 — ER lub RC bez wyjścia OSCOUT

101 — INTRC z wyjściem OSCOUT

100 — INTRC bez wyjścia OSCOUT

011 — EC

010 — HS

001 — XT

000 — LP

bit **WDTE** — bit odblokowania licznika WDT

1 — licznik WDT odblokowany

0 — licznik WDT zablokowany

bit **~PWRTE** — wyłączenie opóźnienia 72 ms po załączeniu zasilania

1 — opóźnienie wyłączone

0 — opóźnienie włączone

bit **MCLRE** — wyłączenie funkcji zewnętrznego zerowania

1 — funkcja zewnętrznego zerowania włączona

0 — funkcja zewnętrznego zerowania wyłączona, udostępnia linie jako wejście

bit **BODEN** — włączenie funkcji zerowania po spadku napięcia zasilania (BOR)

1 — funkcja włączona

0 — funkcja wyłączona

bit **LVP** — włączenie funkcji programowania niskim napięciem

1 — funkcja włączona, zajmuje wybraną linię I/O dla tej funkcji

0 — funkcja wyłączona

bit **WRT** — włączenie funkcji zapisu do pamięci programu

1 — funkcja włączona, zapis do pamięci programu dozwolony

0 — funkcja wyłączona, zapis do pamięci programu zablokowany

bit **CPD** — zablokowanie przed odczytem pamięci danych EEPROM lub FLASH (*Data Protect*)

1 — odczyt pamięci danych możliwy (odblokowany)

0 — odczyt pamięci danych zablokowany

bity **CP1**, **CP0** — bity blokujące odczyt pamięci programu (*Code Protection bits*)

CP1, **CP0** — bity wyboru blokady odczytu pamięci

11 — możliwy odczyt całej pamięci programu

10 — możliwy odczyt dolnej połowy pamięci programu

01 — możliwy odczyt dolnej ćwiartki pamięci programu

00 — cała pamięć programu zablokowana dla odczytu

lub

CP — bit (bity) blokujący odczyt pamięci programu (*Code Protection bits*)

1 — odczyt zawartości pamięci programu możliwy

0 — odczyt zawartości pamięci programu zablokowany

bit **DEBUG** — włączenie układu wewnętrznego debugera (tylko dla PIC16F87X)

1 — układ wewnętrznego debugera wyłączony

0 — układ wewnętrznego debugera włączony, linie RB7 i RB6 używane przez debuger i wyłączone z puli linii I/O

Przed zaprogramowaniem domyślny stan wszystkich bitów konfiguracyjnych jest równy 1.

Dla procesorów *Base-Line* znaczenie bitów jest identyczne, różna jest natomiast szerokość słowa — 12 bitów i położenie ww. słów w pamięci konfiguracyjnej.