Architektura mikrokontrolerów PIC16F8x

część 3

W trzeciej, ostatniej już części krótkiego kursu, przedstawiamy kolejne, istotne dla projektantów systemów mikroprocesorowych na mikrokontrolerach PIC16F, zagadnienia: obsługa przerwań, watchdog, obniżony pobór mocy i możliwe konfiguracje generatora zegarowego.

Zajmijmy się nieco szerzej przerwaniami. Przy okazji omawiania portów, timera/licznika TMR0, pamięci EEPROM oraz rejestru INTCON, zagadnienia te zostały częściowo omówione.

Przypomnijmy więc, że PIC16F84 posiada cztery źródła przerwań:
- przerwanie zewnętrzne (wyprowadzenie RB0/INT),
- przerwanie od przepełnienia timer/licznika TMR0,
- przerwanie od zmian na wyprowadzeniach RB4...RB7 portu B (maski i flagi zgłoszenia od tych przerwań zawarte są w rejestrze INTCON, który pokazano na rys. 5),
- przerwanie generowane w momencie zakończenia operacji wpisu do pamięci EEPROM (flaga zgłoszenia tego przerwania znajduje się w rejestrze EECON1 - rys. 11), natomiast maska w rejestrze INTCON.

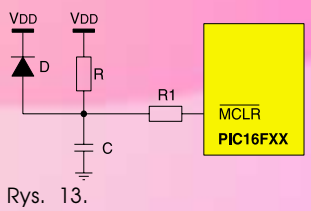
Cały system przerwań może być zablokowany lub odblokowany zależnie od wartości bitu GIE w rejestrze INTCON (rys. 5). Po zerowaniu mikrokontrolera bitu GIE jest również zerowany, co powoduje, że system przerwań jest zablokowany. Każde ze źródeł przerwań może być indywidualnie maskowane.

Kiedy przerwanie jest zgłaszane, bit GIE jest zerowany. Zapobiega to zgłaszaniu dalszych przerwań w trakcie wykonywania procedury obsługi danego przerwania. Na stos zapisywany jest adres powrotu, a do licznika rozkazów wpisywana jest wartość 0004h. Procedura obsługi przerwania musi (rozróżniać poprzez testowanie odpowiednich flag) jakie jest źródło przerwania. Jeżeli flaga jest ustawiona na jeden, to w sekwencji obsługującej ące przerwanie trzeba wyzerować tę flagę. Procedura obsługi przerwania musi kończyć się rozkazem *retfie*. Wykonanie tego rozkazu powoduje ustawienie GIE na 1 i wpisanie ze stosu do licznika rozkazów adresu powrotu. Wśród instrukcji mikrokontrolerów PC16F nie ma rozkazów operacji na stosie typu *push* lub *pop*. W procedurach obsługi przerwania należy zadbać o zachowanie modyfikowanych rejestrów w pamięci RAM. Dotyczy to szczególnie rejestrów W i STATUS.

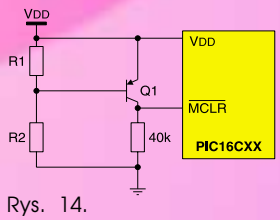
Poniżej pokazano przykład inicjowania i obsługi przerwania od licznika TMR0. Wszystkie inne przerwania są zamaskowane.

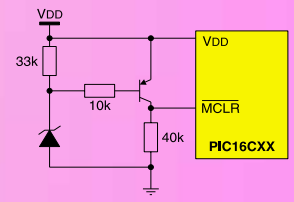
```
  org 0000
  goto _inic
  org 0004
  goto _int
_inic   .    ;tutaj instrukcje
             ;inicjujące TMR0
   .
  clrf INTCON  ;zeruj wszystkie
               ;maski i flagi
  bsf INTCON,T0IE ;odblokowanie
                  ;przerwania od TMR0
  bsf INTCON,GIE  ;odblokowanie
                  ;systemu przerwań
   ...
   ...
_int  movwf w_temp      ;zachowaj
                  ;rejestr W
  movf STATUS,w   ;STATUS do W
  movwf st_temp   ;zachowaj rejestr
                  ;STATUS


  bcf STATUS,RP0 ;ustawienie banku 0
  bcf INTCON,T0IF   ;zerowanie flagi
                ;przerwania od
                ;przepełnienia TMR0
  movlw czas    ;ładuj licznik
                ;(jeżeli przerwanie
                 ;ma być wywoływane
  movwf TMR0    ;sekwencyjnie)
   ...
   ...
  movf st_temp,w
  movwf STATUS   ;odtworzenie STATUS
  movf w_temp,w  ;odtworzenie W
  retfie
```

Mikrokontroler PIC16F84 jest szczególnie odpowiedni do wszelkiego rodzaju układów sterowania. Od takich układów oczekuje się ciągłej pracy, często bez nadzoru i przy narażeniach różnego rodzaju (przepięcia, udary termiczne itp.). Mikrokontroler może być w takich przypadkach narażony na „wytrącenie" z normalnej pracy. Aby, przywrócić go do prawidłowego wykonywania programu sterującego zastosowano układ licznika nadzorującego, tzw. watchdoga WDT. Licznik ten ma swój wewnętrzny

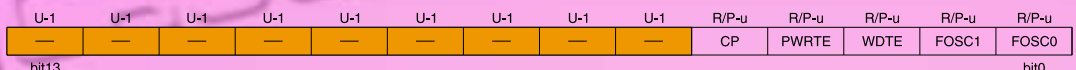


Rys. 13.





Rys. 14.



P-bit programowalny
R-bit który mozna czytać
U-bit niezaimplementowany
u-bit nie zmieniany po włączeniu zasilania

bit 13:5 niezaimplementowany, czytany jako 1'
bit 4 CP: bit zabezpieczenia przed odczytem z pamięci programu i EEPROM użytkownika
1 = brak zabezpieczenia
0 = pamięć zabezpieczona
bit 3 PWRTE: zezwolenie na opóźnienie generowane przez PWRT
1 = opóźnienie odblokowane
0 = opóźnienie zablokowane
bit 2 WDTE: odblokowanie watchdoga
1 = WDT odblokowany
0 = WDT zablokowany
bit 1:0 FOSC1:FOSC0: bity wyboru oscylatora
11 =oscylator RC 10 = oscylator HS 01 = oscylator XT
00 = oscylator LP

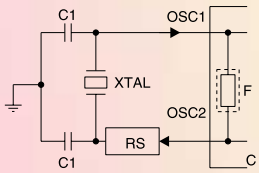
Rys. 15.

generator RC. Czas zliczania wynosi ok. 18ms. Jak wspomniano przy okazji omawiania licznika TMR0, jest możliwe przyłączenie preskalera do WDT. Nie jest wtedy przyłączony do TMR0. Przy maksymalnym stopniu podziału 1:128 można uzyskać czas odmierzania wynoszący ok. 2,3s. Przepełnienie licznika powoduje wygenerowanie sygnału zerującego. Program sterujący powinien być tak napisany, żeby do tego nie dopuścić. Zerowanie licznika odbywa się przez wykonanie instrukcji *clrwdt*.

Napisanie programu, który dobrze wykorzystuje mechanizm watchdoga nie jest sprawą prostą i wymaga pewnej praktyki. Uruchomienie WDT uzyskuje się przez ustawienie bitu (wpisanie jedynki) WDTE specjalnego rejestru konfiguracji CONFIGURATION WORD. Rejestr ten będzie opisany niżej.

Do prawidłowego startu i późniejszej pracy niezbędne jest prawidłowe zerowanie (reset) mikrokontrolera. Jest on wykonywane po włączeniu zasilania mikrokontrolera, podaniu stanu niskiego na wejście MCLR lub w wyniku działania watchdoga WDT.

Impuls zerujący po włączeniu zasilania (POR) jest generowany w momencie, kiedy napięcie zasilania wzrośnie do wartości 1,2...1,7V (wyprowadzenie MCLR musi być połączone z napięciem zasilania bezpośrednio lub przez rezystor). Eliminuje się w ten sposób zewnętrzne elementy RC, zazwyczaj potrzebne do generowania impulsu zerującego. Szybkość narastania napięcia zasilania musi być większa od pewnej minimalnej wartości określonej przez producenta(50mV/ms). Jeżeli napięcie rośnie wolniej, to do generowania impulsu zerowania potrzebne są zewnętrzne elementy RC dołączone do wejścia MCLR. Na **rys. 13** przedstawiony jest zewnętrzny obwód zerujący, zalecany do stosowania dla wolno narastającego napięcia zasilania.



Tryb	Częstotliw.	OSC1/C1	OSC2/C2
LP	32 kHz 200 kHz	68 - 100 pF 15 - 33 pF	68 - 100 pF 15 - 33 pF
XT	100 kHz 2 MHz 4 MHz	100 - 150 pF 15 - 33 pF 15 - 33 pF	100 - 150 pF 15 - 33 pF 15 - 33 pF
HS	4 MHz 10 MHz	15 - 33 pF 15 - 33 pF	15 - 33 pF 15 - 33 pF

Rys. 16.

Przez ustawienie bitu PWRT w CONFIGURATION WORD można odblokować wewnętrzny timer PWRT. Odmierza on opóźnienie ok. 72ms od momentu wygenerowania impulsu POR i generuje impuls PWRT TIME OUT. Od tego momentu startuje kolejne opóźnienie wynoszące 1024 cyklów zegara mikrokontrolera (nie dotyczy to oscylatora RC). Po odliczeniu tego opóźnienia generowany jest właściwy impuls zerujący mikrokontroler. Jak widać proces generowania impulsu zerowania jest dość skomplikowany. Zerowanie musi być zatem dobrze przemyślane, gdyż ma duże znaczenie dla poprawnej pracy mikrokontrolera. Jednym z krytycznych momentów jest chwilowe obniżenie napięcia do wartości minimalnej, ale nie równej zero. Jeżeli jest możliwe wystąpienie takiego przypadku, to trzeba zastosować zewnętrzny obwód zerowania, którego dwa możliwe warianty przedstawiono na **rys. 14.**

W rejestrze STATUS REGISTER są dwa bity - T0 i PD - których wartość określa, jaki rodzaj zerowania ostatnio wystąpił. Jeżeli T0 i PD są jedynkami, to przyczyną ostatniego zerowania było włączenie zasilania (POR) lub na wejściu MCLR pojawił się poziom niski. Jeżeli T0=0 i PD=1, to nastąpiło przepełnienie licznika WDT (watchdog). Kombinacja T0=1 i PD=0 informuje, że wystąpił na wejściu MCLR poziom niski lub wystąpiło przerwanie podczas trwania stanu uśpienia po wykonaniu instrukcji *sleep*. Dla T0=0 i PD=0 nastąpiło „wybudzenie" ze stanu uśpienia przez układ watchdoga.

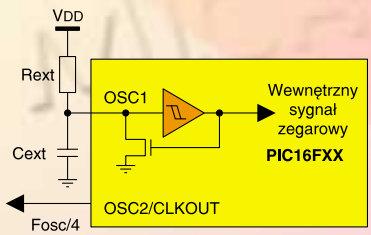
Przy okazji omawiania układu watchdoga oraz układu zerowania wspomnieliśmy o rejestrze konfiguracji CONFIGURATION WORD (**rys. 15**). Jest on umieszczony w specjalnej przestrzeni adresowej pamięci służącej do konfigurowania i testowania (2000h...3FFFh) pod adresem 2007h. Ta przestrzeń jest dostępna tylko dla programatora układu.

Bity FOSC0 i FOSC1 ustalają jedną z czterech możliwych konfiguracji oscylatora taktującego rdzeń mikrokontrolera PIC16F84. Tryby pracy HS, XT i LP dotyczą konfiguracji z rezonatorem kwarcowym lub ceramicznym, jak to pokazano na **rys. 16**. Ponadto, we wszystkich trybach można zamiast rezonatora podłączyć do wejścia OSC1/CLKIN sygnał taktujący z zewnętrznego oscylatora. W trybie RC możliwe jest taktowanie mikrokontrolera poprzez dołączenie do wyprowadzenia OSC1 rezystora i kondensatora, co zilustrowano na **rys. 17**. Zalecane wartości tych elementów mieszczą się w przedziałach: 3kΩ<*Rext*<100kΩ, *Cext*>20pF.

Mikrokontroler PIC16F84 można wprowadzić w stan obniżonego poboru energii za pomocą rozkazu *sleep*. Jeżeli WDT jest uruchomiony, to w momencie wykonania rozkazu *sleep* zostaje wyzerowany, ale nie jest zatrzymywany. Oscylator mikrokontrolera zatrzymuje się, bit PD jest zerowany, a bit T0 jest ustawiany na 1 (STATUS REGISTETR). Wejście MCLR musi być na poziomie wysokim. Wyjście z stanu obniżonego poboru energii następuje w momencie zerowania (zerowanie zewnętrzne przez poziom niski na MCLR lub od watchdoga WDT) lub w momencie przyjęcia przerwania zewnętrznego: od zmian na liniach portu B oraz końca zapisu do pamięci EEPROM. Przerwanie od przepełnienia TMR0 nie „wybudza" mikrokontrolera, ponieważ w trybie *sleep* TMR0 jest zatrzymywany. Dokładny opis działania trybu obniżonego poboru energii zawarty jest w notach katalogowych firmy MICROCHIP.

Na tym można zakończyć opisywanie głównych właściwości mikrokontrolera. Rodzina mikrokontrolerów PIC16Fxx jest tak pomyślana, że programy napisane dla PIC16F84 można stosunkowo łatwo przenosić na inne mikrokontrolery. Łączy je ta sama lista rozkazów i podobny rdzeń. Różnice mogą dotyczyć układów peryferyjnych (rejestrów SFR z nimi związanych), pamięci RAM itp. Pojawiają się też kompilatory języka C - profesjonalne oraz niekomercyjne. Te ostatnie udostępniane w wersji freeware lub za drobną opłatą rejestracyjną. Wszystko to razem może przemawiać za alternatywnym, w stosunku do innych rodzin mikrokontrolerów, wykorzystaniem PIC-ów. Oczywiście, Microchip wzbogaca swoją ofertę o inne rodziny mikrokontrolerów. Przykładem są układy PIC17xx i - ostatnio szeroko reklamowane - PIC18Cxx, bardzo szybkie i bogato wyposażone mikrokontrolery.

Tomasz Jabłoński, AVT
tomasz.jablonski@ep.com.pl



Rys. 17.